

ESCOLA POLITÉCNICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

ILDEVANA POLTRONIERI RODRIGUES

USA-DSL PROCESS: A USABILITY EVALUATION PROCESS FOR DOMAIN-SPECIFIC LANGUAGES

Porto Alegre
2021

PÓS-GRADUAÇÃO - *STRICTO SENSU*



Pontifícia Universidade Católica
do Rio Grande do Sul

**PONTIFICAL CATHOLIC UNIVERSITY OF RIO GRANDE DO SUL
SCHOOL OF TECHNOLOGY
COMPUTER SCIENCE GRADUATE PROGRAM**

**USA-DSL PROCESS: A
USABILITY EVALUATION
PROCESS FOR
DOMAIN-SPECIFIC
LANGUAGES**

ILDEVANA POLTRONIERI RODRIGUES

Thesis submitted to the Pontifical Catholic
University of Rio Grande do Sul in partial
fulfillment of the requirements for the
degree of Ph. D. in Computer Science.

Advisor: Prof. Dr. Avelino Francisco Zorzo
Co-Advisor: Prof. Dr. Maicon Bernardino da Silveira

**Porto Alegre
2021**

Ficha Catalográfica

R696u Rodrigues, Ildevana Poltronieri

Usa-DSL Process : A Usability Evaluation Process for
Domain-Specific Languages / Ildevana Poltronieri Rodrigues. –
2021.

174 p.

Tese (Doutorado) – Programa de Pós-Graduação em Ciência da
Computação, PUCRS.

Orientador: Prof. Dr. Avelino Francisco Zorzo.

Co-orientador: Prof. Dr. Maicon Bernardino Silveira.

1. Usability. 2. Evaluation. 3. Process. 4. Domain-Specific Language.
5. DSL. I. Zorzo, Avelino Francisco. II. Silveira, Maicon Bernardino.
III. Título.

Elaborada pelo Sistema de Geração Automática de Ficha Catalográfica da PUCRS
com os dados fornecidos pelo(a) autor(a).

Bibliotecária responsável: Clarissa Jesinska Selbach CRB-10/2051

Ildevana Poltronieri Rodrigues

Usa-DSL Process: A Usability Evaluation Process for Domain-Specific Languages

This Thesis has been submitted in partial fulfillment of the requirements for the degree of Doctor of Computer Science, of the Graduate Program in Computer Science, School of Technology of the Pontifícia Universidade Católica do Rio Grande do Sul.

Sanctioned on 26 August 2021.

COMMITTEE MEMBERS:

Prof. Dra. Isabela Gasparini (PPGCA/UDESC)

Prof. Dr. Marcelo Soares Pimenta (PPGC/UFRGS)

Prof. Dr. Luiz Gustavo Leão Fernandes (PPGCC/PUCRS)

Prof. Dr. Avelino Francisco Zorzo (PPGCC/PUCRS - Advisor)

Prof. Dr. Maicon Bernardino da Silveira (PPGES/UNIPAMPA – Co-Advisor)

*To my lovely daughter Ana Carolina, my husband Elder, and my mother Ana Florinda,
who offered me unconditional support during the course of this thesis.*

“I will say of the Lord, “He is my refuge and my
fortress, my God, in whom I trust.”
(Psalm 91:2)

ACKNOWLEDGMENTS

Thank God for giving me health and guiding my decisions.

Thank my daughter for her understanding of the times I was not there and I did not give her the attention she deserves.

I am immensely grateful to my husband for never letting me give up, however difficult it was.

To my mother for her prayers, words of optimism and unconditional support at all times.

To my mother-in-law and father-in-law for the cheering and words of support in this trajectory.

To my Advisor Avelino for the trust placed in my decisions, for the guidance and for the understanding at all times of this long journey.

To my co-advisor Maicon for his constant help in this doctoral research.

To the student Allan for agreeing to dive right into this project.

To all students who were part of this work in any way, whether in the construction, evaluation or participation in the evaluations.

I thank all the researchers who contributed to making this work possible I will mention some who were closest to Edson, Jean, Amanda, Elder, and Márcia.

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001.

USA-DSL PROCESS: UM PROCESSO DE AVALIAÇÃO DE USABILIDADE PARA LINGUAGENS ESPECÍFICAS DE DOMÍNIO

RESUMO

Atualmente, existem diferentes Linguagens Específicas de Domínio (Domain-Specific Languages - DSL) que são criadas com o objetivo de resolver problemas em um domínio particular e não se destinam a resolver problemas fora de seu âmbito de atuação. De acordo com Fowle uma DSL é uma linguagem computacional focada em um domínio particular, no entanto sua expressividade está limitada a um domínio específico. Estas linguagens estão classificadas conforme sua forma de implementação, podendo ser internas, externas e gráficas. Sendo assim, a partir das diferentes aplicações e classificações das DSLs, uma preocupação tem sido pertinente: como avaliar a usabilidade das linguagens desenvolvidas. Quanto a avaliação destas linguagens, pode-se encontrar diversos estudos experimentais avaliando subjetivamente usabilidade, mas poucos se apropriaram das técnicas específicas de Interação Humano-Computador (IHC). Sendo assim, busca-se com esta tese identificar o estado da arte no que se refere a processo de avaliação de usabilidade para DSL. Com base em um *framework* de avaliação de usabilidade para DSL e estudos encontrados na literatura, esta tese apresenta um processo de avaliação de usabilidade para DSLs, que destina-se a avaliação centrada no usuário.

Palavras-Chave: Processo de Avaliação, Usabilidade, Interação-Humano Computador, Engenharia de Software, DSL, Teste de Desempenho.

USA-DSL PROCESS: A USABILITY EVALUATION PROCESS FOR DOMAIN-SPECIFIC LANGUAGES

ABSTRACT

Currently, there are different Domain-Specific Languages (DSL) that are created to solve problems in a particular domain and are not intended to solve problems outside their scope. According to Fowler, a DSL is a computational language focused on a particular domain, however, its expressiveness is limited to that specific domain. These languages are classified according to their form of implementation and can be internal, external, and graphic. Thus, from the different applications and classifications of DSLs, a concern has been pertinent: the usability evaluation the developed languages. As for the evaluation of these languages, several experimental studies can be found subjectively evaluating usability, but few have appropriated the specific techniques of Human-Computer Interaction (IHC). Therefore, this thesis seeks to identify the state of the art regarding the usability evaluation process for DSL. Based on a framework of usability evaluation for DSL and the studies found in the literature, we developed a usability evaluation process for DSLs, which is user-centric.

Keywords: Evaluation Process, Usability, Human-Computer Iteration, Software Engineering, DSL, Performance Testing.

LIST OF FIGURES

1.1	Research Methodology	18
1.2	Research Design	19
2.1	Textual DSL Example (Textual Scenario PerfText)	23
2.2	Textual DSL Example (Textual Script PerfText)	23
2.3	Graphical DSL Example (Graphical Script Canopus [14])	24
2.4	Graphical DSL Example (Graphical Scenario Canopus [14])	24
2.5	PEAR phases steps/activities order (BPMN notation)	31
2.6	Usa-DSL Framework	32
2.7	Method Framework SPEM	36
2.8	BPMN Elements	42
3.1	Search String	45
3.2	Studies Selection Process	47
3.3	DSL Usability Evaluation Taxonomy	56
4.1	Usa-DSL Process Fundamental Principles	62
4.2	SPEM/Usa-DSL Mapping	63
4.3	Profiles	64
4.4	The Process - Step 1	65
4.5	Method Content	66
4.6	Process	66
4.7	Usa-DSL Process Structure	68
4.8	Usa-DSL Process Flow BPMN	70
4.9	Activity P5 - Task P5a Flow	71
4.10	Task P5a (expanded)	71
4.11	Mapping SPEM/Usa-DSL	73
4.12	Work Products - Guidance	74
4.13	Methodology	75
4.14	CDN and Taxonomy Metrics	80
5.1	Perceived Usefulness Results of Usa-DSL	87
5.2	Ease of Use Results of Usa-DSL	88
7.1	Usability Test Workflow	104
7.2	Heuristic Evaluation Workflow	110
8.1	Usa-DSL Tool (Interface Sample)	119

A.1	Step 1 - Evaluators Profiles	128
A.2	Step 2 - Ethical and Legal Responsibilities	128
A.3	Step 3 - Data Type	129
A.4	Step 4 - Empirical Study Method (SE)	129
A.5	Step 5 - Evaluation Method (HCI)	129
A.6	Step 6 - Metrics	130
A.7	Step 7 - Gathering Instruments	130
A.8	Step 8 - Evaluation Instructions	130
A.9	Step 9 - Evaluation Conduction	131
A.10	Step 10 - Data Packaging	131
A.11	Step 11 - Evaluation Reporting	131
C.1	Task P1a - Choose the Profile Evaluator	134
C.2	Tasks of E1 - Apply Instruments to Identify Profiles	134
C.3	Task A1a - Analysis of Profile Subjects	135
C.4	Tasks of R1 - Report Evaluator Profiles	135
C.5	Task P2a - Select the Informed Consent Term	136
C.6	Tasks of E2 - Introduce the Form and Collects Signatures of Subject	136
C.7	Tasks of R2 - Report Subjects Number and Used Form	137
C.8	Task P3a - Choose the Data Type	137
C.9	Task P4a - Select the Empirical Study Method	137
C.10	Tasks of E4 - Develop and Conduct Protocol	138
C.11	Task A4a - Analyze Activities of the Study Protocol	138
C.12	Tasks of R4 - Report the Developed Protocol	138
C.13	Task P5a - Select the Evaluation Usability	139
C.14	Tasks of E5 - Prepare the Evaluation	139
C.15	Tasks of R5 - Report Conduction Evaluation	140
C.16	Task P6a - Choose the Metrics	140
C.17	Task P7a - Select the Instruments Data Gathering	141
C.18	Tasks of E7 - Data Collection	141
C.19	Tasks of A7 - Analyze the Collected Data	141
C.20	Tasks of R7 - Report Analysis	142
C.21	Task P8a - Select the Instruments Training	142
C.22	Tasks of E8 - Introduce Instruments of Instruction and Conduct Training	143
C.23	Tasks of R8 - Report Data Analysis	143

C.24 Task P9a - Evaluation Place Definition 143

C.25 Tasks of E9 - Execution and Evaluation Conduction 144

C.26 Tasks of A9 - Analyze the Performed Tasks 144

C.27 Tasks R9a - Describe the Evaluation Following the Chosen Methods 144

C.28 Task P10a - Choose the Data Storage 145

C.29 Tasks of E10 - Store Data Obtained 145

C.30 Tasks P11a - Select the Study Reporting 146

C.31 Task A11a - Analyze the Documentation to be Used in the Evaluation 146

C.32 Tasks of R11 - Report the Results and Analyzed Information 146

LIST OF TABLES

3.1	Specific String	45
3.2	Quality Assessment	46
3.3	Importance of Usability	50
3.4	Evaluation Data	51
3.5	DSL Constraints and Evaluation Constraints	52
4.1	Mapping Usa-DSL Process to SPEM	72
4.2	Severity Degree.	77
4.3	Snippet for our HEC	78
6.1	Subjects Profile	95

CONTENTS

1	INTRODUCTION	15
1.1	PROBLEM STATEMENT AND RATIONALE FOR THE RESEARCH	16
1.2	OBJECTIVES	17
1.3	THESIS RELEVANCE	17
1.4	RESEARCH METHODOLOGY	18
1.5	ORGANIZATION	20
I	Literature Review	21
2	BACKGROUND	22
2.1	DOMAIN-SPECIFIC LANGUAGES	22
2.2	USABILITY ENGINEERING	24
2.3	HEURISTIC EVALUATION	26
2.4	COGNITIVE DIMENSIONS	28
2.5	USA-DSL FRAMEWORK	30
2.6	SOFTWARE PROCESS	35
2.7	SOFTWARE AND SYSTEMS PROCESS ENGINEERING METAMODEL	35
2.7.1	SPEM ELEMENTS	36
2.8	BUSINESS PROCESS MODEL NOTATION	38
2.8.1	BPMN ELEMENTS	40
2.9	FINAL CONSIDERATIONS	42
3	SYSTEMATIC LITERATURE REVIEW	44
3.1	SLR PLANNING	44
3.2	SLR EXECUTION	45
3.3	SLR ANALYSIS AND ANSWERS TO RESEARCH QUESTIONS	47
3.4	EVOLUTION OF USABILITY EVALUATION FOR DSL	54
3.5	EVOLUTION TAXONOMY FOR DSL EVALUATION	55
3.6	FINAL CONSIDERATIONS	58

II Usa-DSL Process Conception 60

4	USA-DSL PROCESS	61
4.1	METHOD CONTENT - USA-DSL PROCESS	63
4.2	PROCESS - USA-DSL PROCESS	65
4.3	USA-DSL PROCESS STRUCTURE	68
4.4	USA-DSL PROCESS LIFE CYCLE MODELING BPMN	68
4.5	MAPPING USA-DSL <i>PROCESS</i> TO SPEM	71
4.6	USA-DSL PROCESS WORK PRODUCTS	74
4.6.1	HEURISTIC EVALUATION CHECKLIST FOR DSL	75
4.6.2	USABILITY QUESTIONNAIRE	77
4.6.3	METRICS AND COGNITIVE DIMENSIONS MAPPING	78
4.6.4	PROFILE QUESTIONNAIRE	80
4.6.5	INFORMED CONSENT TERM	81
4.7	FINAL CONSIDERATIONS	83

III Usa-DSL Process Evaluation 84

5	USA-DSL EVALUATION: SURVEY	85
5.1	PLANNING	85
5.2	PILOT STUDY	85
5.3	RESULT ANALYSIS	86
5.4	USA-DSL PROCESS EVOLUTION	91
5.5	THREATS TO VALIDITY	91
5.6	FINAL CONSIDERATIONS	92
6	HEURISTIC EVALUATION CHECKLIST: INTERVIEW	93
6.1	INTERVIEW CONTEXT	93
6.2	PILOT TEST	94
6.3	PROFILE	94
6.4	INTERVIEWS	95
6.4.1	QUALITATIVE ANALYSIS	96
6.4.2	DISCUSSION	98
6.5	EXAMPLE OF USE	99
6.5.1	ANALYSIS	99

6.6	FINAL CONSIDERATIONS	101
IV	Usa-DSL Process Example of Use	102
7	USA-DSL PROCESS: EXAMPLE OF USE	103
7.1	STUDY 1: USABILITY TEST	103
7.1.1	USABILITY TEST DEFINITION	106
7.2	STUDY 2: HEURISTIC EVALUATION	109
7.2.1	HEURISTIC EVALUATION DEFINITION	109
7.3	PROCESS EXECUTOR OPINION - USA-DSL PROCESS	113
7.4	FINAL CONSIDERATIONS	115
8	THESIS SUMMARY AND FUTURE WORK	116
8.1	THESIS CONTRIBUTIONS	116
8.2	LIMITATION AND FUTURE WORK	118
8.3	PUBLICATIONS	120
	REFERENCES	121
	APPENDIX A – Steps - Usa-DSL Process	128
	APPENDIX B – Activities - Usa-DSL Process	132
	APPENDIX C – Tasks - Usa-DSL Process	134
	APPENDIX D – Heuristic Evaluation Checklist for Graphical and Textual DSL - Usa-DSL	147
	APPENDIX E – Usability Questionnaire for Graphical and Textual DSL - Usa-DSL (Extrated and Adapted the Cognitive Dimensions Questionnaire)	157
	APPENDIX F – Usability Questionnaire for Graphical and Textual DSL - Usa-DSL Version Adapted	164
	APPENDIX G – Informed Consent Term	172
	APPENDIX H – Profile Questionnaire	173

1. INTRODUCTION

Currently, Computer Science permeates different fields of knowledge in which it is used to investigate and solve problems through computational systems in different application domains. Several computational systems have been applied to a myriad of areas of knowledge, from Social Science to Engineering. Given the diversity of domains, the development of different languages to be applied to these various areas has increased, *i.e.*, Domain-Specific Languages (DSL) [48]. Despite the increase in the number of languages and the effort needed to develop them, it is possible to verify that many are unsuccessful and end up falling into disuse. One of the possible causes might be due to the fact that usability is not being properly considered/evaluated. In such cases, usability refers to a set of methods that allow a user to understand and identify elements that facilitate the use of a language [74].

In this sense, one of the subfields of Computer Science, *i.e.*, Human-Computer Interaction (HCI), is concerned with “the design, evaluation, and implementation of interactive computer systems for human use and with the study of the major phenomenon surrounding them” [36]. In this regard, the HCI area [67] aims to create and evaluate systems that support its users. However, some issues, such as domain knowledge and the languages to be evaluated, cannot be solved solely with HCI support. In order to solve these problems, it is possible to use techniques or methods developed in Software Engineering (another subfield of Computer Science).

Software Engineering (SE) [80] is concerned with the ability of the software product to be understood, learned, operated by the user, and still complying with guidelines and requirements when used under specific conditions. In the context of DSL development, Language Engineers should consider that users need to have knowledge in the system’s domain but, at the same time, without the need to be specialists in Computer Science. Bearing that in mind, domain users should be able to use the developed languages more independently and as easily as possible through the design of DSLs. To that end, Language Engineers should be concerned not only with usability criteria but also with the diversity of contexts and domains, taking into consideration the satisfaction in use during the interaction of these users with different systems. In order to mitigate the growing concern with the usability of such languages [70], some frameworks for evaluating the usability of DSL [63] [12] were developed. For instance, the framework provided by Barisic *et al.* [12], despite presenting empirical studies such as experiments and surveys, does not present or address HCI concepts and artifacts, evaluating usability exclusively through usability metrics. The framework proposed by Poltronieri *et al.* [62] also supports usability evaluation taking into account empirical studies, but with an emphasis on HCI concepts, specifically those of usability evaluation, *i.e.*, Usability Test and Heuristic Evaluation. However, despite the Usa-DSL framework having been developed and having appropriate HCI concepts, the need for

the framework to be guided by a well-defined process was identified while it was evaluated. For this purpose, it was necessary to define a process that encompasses a set defined by good practices that can be reproduced. Furthermore, it is crucial that the process presents predictable and detailed steps, workflows, roles, activities, and tasks, in addition to the support of guidelines and artifacts used by the tasks performed during the assessment [62]. Although Software Engineering provides several processes that present activities to obtain a system so as to meet users' needs, to the best of our knowledge, we have not yet identified any process that specifically guides a DSL usability evaluation [60]. Therefore, in the context of this thesis, a usability evaluation process will be developed. The primary goal of our process is to facilitate the usability evaluation as well as prevent Language Engineers from needing to deepen their knowledge in usability methods. Basically, our process, called Usa-DSL Process, was developed using Software and Systems Process Engineering Metamodel (SPEM) [58], Business Process Model and Notation (BPMN) [59], and the Usa-DSL framework [62]. In addition to its development, evaluations were carried out in different contexts to verify the behavior of the process as a whole in order to ensure that it would be correctly executed by the stakeholders.

1.1 Problem Statement and Rationale for the Research

When developing a Domain-Specific Language (DSL), Language Engineers seek to facilitate the understanding and communication among users and language. To achieve that, it is essential that Language Engineers have a set of well-defined practices, that is, an evaluation process that helps them to evaluate the usability of the developed DSLs.

In order to provide the before-mentioned set of practices, previous studies developed a usability evaluation framework for DSLs, called Usa-DSL Framework [63] (see Chapter 2). Aiming at assessing this framework, a focus group and interviews were used [62]. Although such assessments show evidence of their efficiency, they indicated the need to develop a process to guide their application and the elaboration of artifacts to be used along the process.

Thus, the focus of this thesis is: “The application of a specific process to help Language Engineers¹ to evaluate developed DSLs, making the elaboration of evaluation protocols and the use of artifacts easier, with less mental effort (cognitive load), and fostering the learning of methodologies. Consequently, it facilitates planning, execution, analysis, and reporting of usability evaluations for DSLs.”

From this statement, we sought to answer the following research question:

¹Language Engineers: in the context of Usa-DSL Process, they are composed of domain analysts, developers, and testers.

Research Question: “Can a usability evaluation process for DSL help Language Engineers to plan, execute, analyze, and report their evaluation?”

To answer this research question, a Usability Evaluation Process for DSL was developed, as well as the artifacts that support this process. In order to verify whether this process was adequate, some assessments were carried out, that is, interviews and surveys.

1.2 Objectives

In this section, the general and specific objectives of this thesis will be presented.

General Objective: The general objective of this work is to develop a Usability Evaluation Process for DSLs supported by the Usa-DSL Framework.

Specific Objective: To achieve the general objective of this study, the following specific objectives were established:

- to identify different concepts for developing the process and its artifacts;
- to propose a process that uses the concepts defined in the Usa-DSL framework;
- to develop the artifacts that will be used in the usability evaluation process;
- to evaluate the developed artifacts through interviews with HCI experts;
- to evaluate the developed process through an empirical study.

1.3 Thesis Relevance

This research is of particular relevance to the scientific community since: (i) it develops a process that supports the usability evaluation of the developed DSLs to meet research or industrial needs; (ii) for software development companies, the Usa-DSL Process can help Language Engineers to evaluate the usability of their developed DSL; (iii) both academia and industry can benefit from this study by obtaining information about the use of knowledge in the field of Human-Computer Interaction and Software Engineering to facilitate and to improve the process of evaluating the usability of DSLs.

In addition to the previous contributions, we can also highlight the detailing of the Usa-DSL Process as support for Language Engineers to achieve independence while developing a usability evaluation. It also gives freedom for researchers to use the usability evaluation methods in the Human-Computer Interaction field with more property and less effort in terms of learning the methods. Usa-DSL has a graphical presentation of the workflow,

life cycle, and elements that compose it, as well as descriptions on how to plan, execute, analyze, and report the conducted evaluations.

1.4 Research Methodology

Before we present the methodology used in this thesis, it is important to understand the context of our research. Basically, our research is divided into three parts: 1) the design of a usability evaluation framework, called Usa- DSL framework, which was previously published [62]; 2) the development of a usability evaluation process, called Usa-DSL Process [71], which will be described in this thesis and follows the Usa-DSL Framework; and, 3) the implementation of a tool to assist the design of DSL usability evaluation, which will follow the Usa-DSL Process. Each part will be briefly described next:

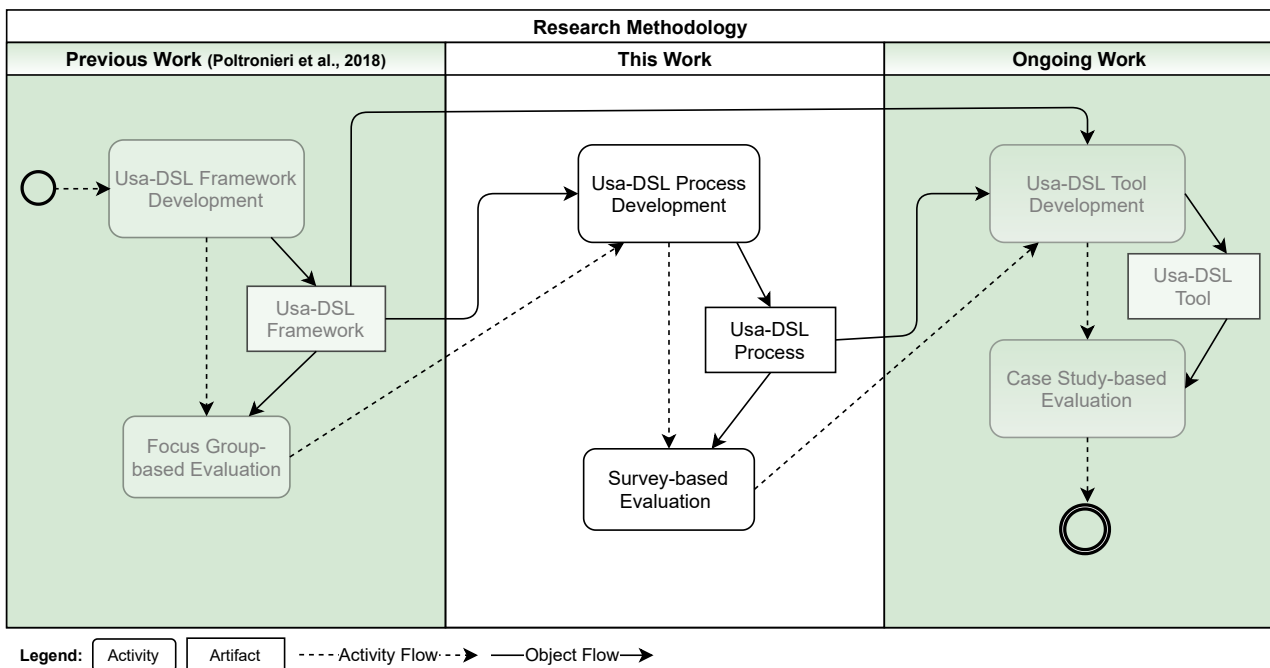


Figure 1.1: Research Methodology

Previous Work: our previous work presented the main motivation for the development of the Usa-DSL framework, *i.e.*, the need to evaluate a DSL created for performance testing. We will briefly describe the Usa-DSL Framework in Section 2.5. The main aspects described in our previous work are related to the framework concepts and requirements, besides a focus group evaluation. One of the aspects pointed out by the focus group was the need to provide an execution flow (process) for the DSL designer. This process would help DSL designers to understand the order of the tasks they would have to execute when building the usability evaluation of their DSL.

This Work: this work will present the Usa-DSL Process and a survey-based evaluation. It is worth noting that our process is based on the Usa-DSL Framework built in our

previous work. Aspects related to “Who will do what, when, and how?” will be presented in Section 4.2.

Ongoing Work: in order to speed up the process of building the needed artifacts described in our process, we are implementing a tool to help DSL designers build their usability evaluation. The tool, called Usa-DSL Tool, will generate all artifacts based on templates, such as invite e-mails, questionnaires, informed consent terms, and so on. This tool will not be presented in this thesis.

Once the scope of this thesis has been defined, we present the methodology used in this investigation, which is based on the objectives defined in Section 1.2.

The methodology of this thesis follows the research design shown in Figure 1.2. For its development, we planned the research methodology organized in four parts: Usa-DSL Process Literature Review, Usa-DSL Conception, Usa-DSL Process Evaluation, and Usa-DSL Process Example of Use. These parts will be described in detail as follows:

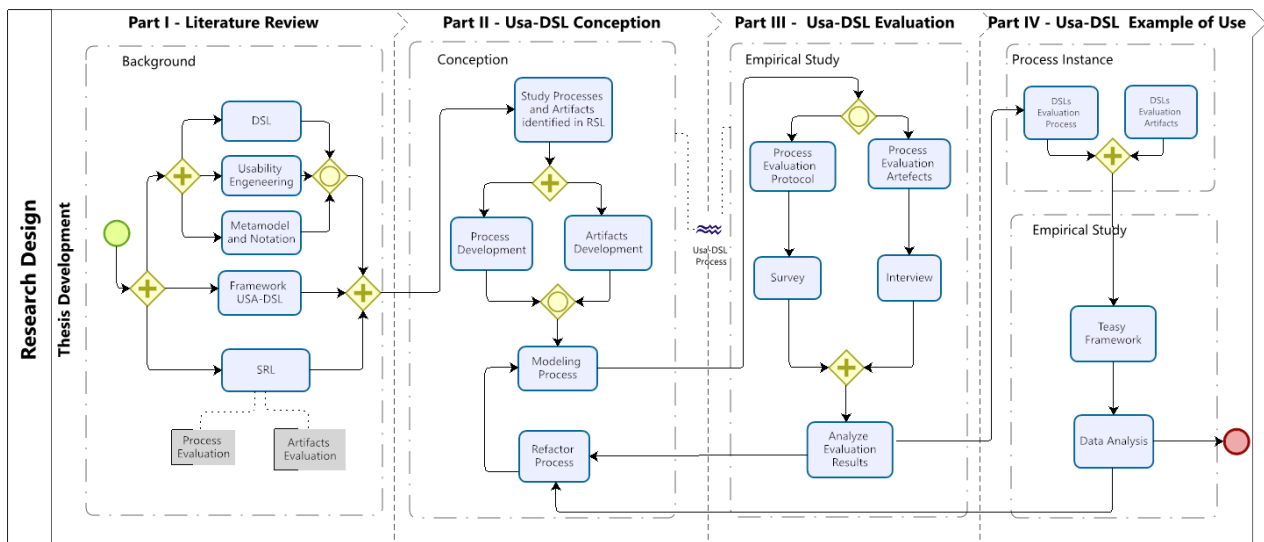


Figure 1.2: Research Design

1. Thesis Development: This step comprises the blocks background, conception, empirical study and process instance.

Part I - Literature Review: this part presents the theoretical basis of this thesis, including the definition of the idea, as well as the theoretical investigation regarding Domain-Specific Language (DSL), Usability Engineering, Metamodel, Model, and Usa-DSL framework. This part also presents a Systematic Literature Review, which aims to establish the theoretical basis for the main research topics, Usability Evaluation Process and Process Artifacts.

Part II - Usa-DSL Process Conception: the activities of this part are presented in the conception block and have the function of studying the different Processes and Artifacts, as well as identifying, analyzing, and developing the process and specific artifacts to

evaluate the usability of DSL. The design part contemplates the development of the product generated from the Usa-DSL framework, which is the Usa-DSL Evaluation Process.

Part III - Usa-DSL Process Evaluation: the third part consists of a single block called Empirical Study. This block includes two studies carried out with the aim of evaluating the process and the artifacts developed. In this part, we defined that the process evaluation would be carried out through a survey and the artifact through an interview. From the analysis of the study data, the process will be refactored, generating a new version.

Part IV - Usa-DSL Process Example of Use: the fourth part is divided into two blocks, namely, Process Instance and Empirical Study. The first block includes the definition of instances of the Usa-DSL process and the definition of artifacts, specific to Teasy's evaluation. The last block defines the example of use that will be developed in partnership with the Laboratory of Empirical Studies in Software Engineering (LESSE), in which a different DSL is being developed. The example of use will be carried out to evaluate the usability of the Teasy framework using the Heuristic Evaluation and Usability Test instances. At the end execution of the example of use (Study 1 and Study 2), the perception of the Language Engineer regarding the use of the Usa-DSL Process during the Studies will be raised.

1.5 Organization

This thesis is organized as follows: **Chapter 2** presents the background information about Domains-Specific Language, Usability engineering, Usa-DSL Framework, Software Process, Software and Systems Process, Engineering Metamodel, and Business Process Model Notation. **Chapter 3** presents a Systematic Literature Review on Usability Evaluation Process, which provides a broad overview regarding the relevant contributions on Usability Evaluation, in particular on DSL. **Chapter 4** presents our Process called **Usa** for Usability and **DSL** for Domain-Specific Language, as well as, the description and development of all elements and documents which are used during its application. **Chapter 5** presents the Usa-DSL Process evaluation, which consists of a survey to verify the users' perception regarding Perceived Usefulness and Ease of use. **Chapter 6** presents the evaluation of the Heuristic Checklist, which consists of a set of interviews to collect and understand the vision of specialists in heuristic evaluation when analyzing and using the checklist to carry out an evaluation. **Chapter 7** presents two examples of use where the Usa-DSL was applied, that is, language designers used Usa-DSL to evaluate their DSLs with the support of the researcher this thesis. Finally, conclusions and future work directions are presented in **Chapter 8**.

Part I

Literature Review

2. BACKGROUND

This chapter presents the basic concepts related to this thesis. Section 2.1 briefly introduces the definition, origin, appearance, and implementation of Domain-Specific Language. Section 2.2 define Usability Engineering. Section 2.3 presents the Heuristic Evaluation method and Section 2.4 explains the Cognitive Dimensions Framework, which we apply/adapt to some of the artifacts for our process. Finally, Section 2.5 provides an overview of the Usa-DSL Framework, which is the base for the development of our process.

2.1 Domain-Specific Languages

A Domain-Specific Language (DSL) is a language with a special purpose or specialized to a particular application domain. According to Van Deursen *et al.* [85] a DSL is “a programming language or executable specification language that offers, through appropriate abstractions, focused expressive power and usually it is restricted to a specific problem domain”. Like other languages, DSLs must have a set of sentences well known by their own syntax and semantics. Fowler [27] asseverates that a DSL is defined as “a computer programming language with limited expressiveness and focused on a particular problem domain”.

The application of DSLs allows the software to be developed faster and more effectively. The major advantage observed in using a DSL is that the knowledge required for its applicability is abstracted to another level. In this way, domain experts can understand, validate, and modify code, tailoring the model to their needs, making the impact of change easier to understand. There is still a significant increase in productivity, reliability, ease of use, and flexibility [85]. According to Mernik [48], DSLs can be classified under three different dimensions: **origin**, **appearance**, and **implementation**.

Regarding the **origin** of a DSL, it can be internal and external. An **internal** DSL is designed from the syntactic and semantic rules of grammar of an existing language, which may be a general-purpose language or another DSL. An **external** DSL is a distinct syntax language that relies on its own infrastructure for lexical, syntactic, semantic analysis, interpretation, compilation, optimization, and code generation.

With regard to the **appearance** dimension, a DSL can be classified as **textual**, **graphical**, **tabular**, and **symbolic**. When in textual format (see Figure 2.1 and Figure 2.2), a DSL allows the domain to be expressed with characters, which are then combined to generate words, expressions, sentences, and instructions that follow the grammar rules previously established in the language. Non-textual DSLs follow the same logic (see Figure 2.4 and

Figure 2.3), but using graphical elements to allow the user to express the domain knowledge with a higher level of understanding and using symbols, tables, figures, and connectors.

As far as the **implementation** dimension is concerned, DSLs can be classified from the perspective of their implementation. These classifications form four groups: (i) well-known execution DSL; (ii) DSL that serves as input to application generators; (iii) non-executable DSLs but useful as input to application generators; (iv) DSL not designed to be executed.

In general, the main consideration for building a DSL should be its origin as each approach has specific advantages and disadvantages that are inherent in each type [27]. Although external DSLs may have an effort associated with building it many times higher than that of an internal DSL, there are currently tools that support DSL building. These tools are known as **Language Workbenches (LWB)** and apply language-oriented programming concepts, providing a higher level of abstraction for complex infrastructure issues [26].

```

1 Scenario: Evaluate the loadtesting workload for 1000 users simultaneously
2   Given 100 users enter the system for each 00:01:00
3   And 200 users leave the system for each 00:00:30
4   And 1000 users register into the system simultaneously
5   And performance testing execution during 04:00:00
6   When 60 percent of the virtual users execute the #Browsing# user profile:
7     Then 60 percent of them execute TPCWShop script
8     And 25 percent of them execute TPCWShop script
9     And 15 percent of them execute TPCWShop script
10  When 15 percent of the virtual users execute the #Ordering# user profile:
11    Then 20 percent of them execute TPCWShop script
12    And 30 percent of them execute TPCWShop script
13    And 50 percent of them execute TPCWShop script
14  When 25 of the virtual users execute the #Shopping# user profile:
15    Then 25 percent of them execute the Browser script
16    And 45 of them execute the TPCWShop script
17    And 30 of them execute the TPCWShop script

```

Figure 2.1: Textual DSL Example (Textual Scenario PerfText)

```

1 include 'data.csv' as parameter
2 Script: the users performs purchase interactions based on TestCaseN1 from TPCWShop
3   Given the HomePage transaction activity through "http://localhost:8080/tpcw/ TPCW_home_interaction" action,
4     which is loaded in 5.0 seconds
5   When i click on "Subject" link parameter.'column'. 'line' which is dynamically generated and update on Each
6     Interaction based on a random strategy
7   Then i will be taken to "/TPCW_new_products_servlet" action in the NewProduct transaction activity, which is
8     loaded in 5.0 seconds
9   And i click on "Product" link parameter.'column'. 'line' which is dynamically generated and update on Each
10    Interaction based on a random strategy
11  And i need to wait thinking time during 00:00:05
12  Then i will be taken to "/TPCW_product_detail_servlet" action in the ProductDetails transaction activity, which
13    is loaded in 3.0 seconds

```

Figure 2.2: Textual DSL Example (Textual Script PerfText)

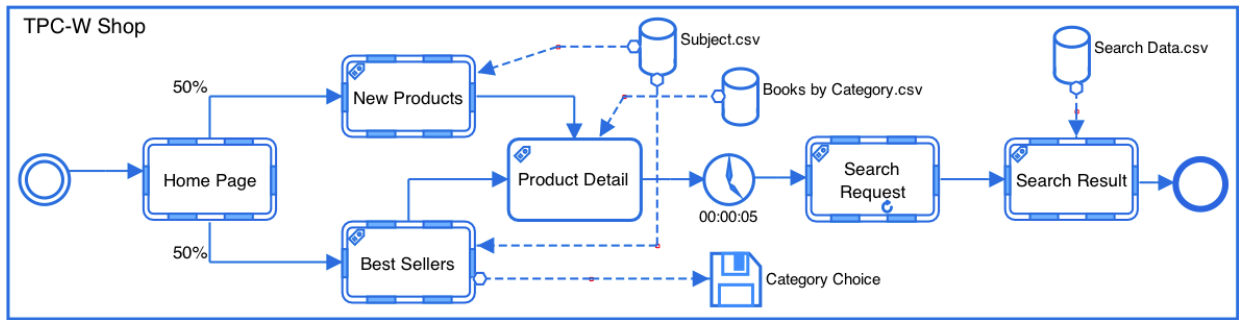


Figure 2.3: Graphical DSL Example (Graphical Script Canopus [14])

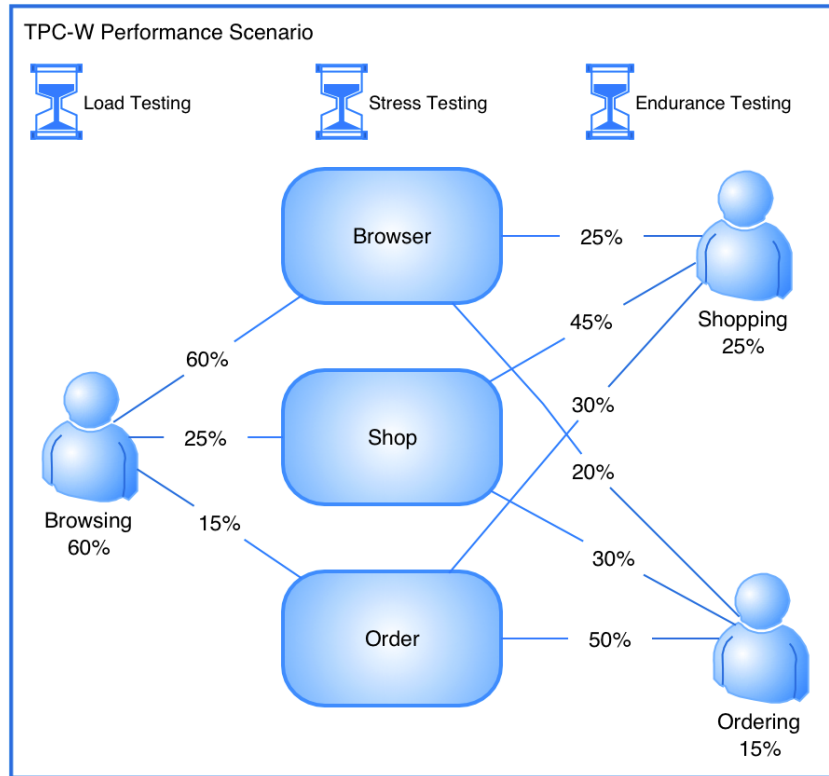


Figure 2.4: Graphical DSL Example (Graphical Scenario Canopus [14])

2.2 Usability Engineering

Usability Engineering (UE) is a discipline that focuses on improving the usability of interactive systems. It draws on theories from Computer Science and Psychology to define problems that occur during the use of such a system. This discipline, in addition to representing the application of techniques, processes, methods, and procedures for the elaboration and evaluation systems and products usable, also considers the importance of human factors involved in software projects [55] [54].

The benefits achieved by the application of UE techniques are visible both in the aspect of efficiency and effectiveness of the interface and are also expressed in more productive systems development processes, reliable and with a high productivity user and

customer satisfaction [55]. When reutilized to ensure a high level of usability on the final interface of computer software, the UE directs to the development of products with a focus on usability.

One of the main goals of Usability Engineering is to improve the usability of interactive systems. Interactive systems for Human-Computer Interaction (HCI) are manipulated programs used by people/users, with an interaction between the user and the computer. However, to succeed in this interaction, it is necessary to verify the way the software components are modeled. These components are composed of: an application that is the non-interactive part of the system, such as functions and business rules, and the interface that is the interactive part, in which the user will interact from actions performed in an interface [73].

Thinking about the development of these interactive systems, we must try to understand which people would use it, which is not always an easy task, as putting yourself in the user's place when we know the domain, allows us to develop a system with fewer errors. However, when you do not have the necessary information about this user, or we are not aware of their needs, it becomes a challenging task. Therefore, one must always take into account **who**, **where**, and **how** the user intends to use this system. Based on this information, the use of the interaction design process is recommended, this process is composed of four basic activities: establishing requirements, creating design alternatives, prototyping and evaluating [73]. It should be noted that this is an iterative and incremental process, and after evaluating the system and analyzing the generated results, it may be necessary to return to the re-design activity [56].

Another important aspect about the interactive design process is the use of other sources of knowledge to develop the research, including paradigms, theories, models, processes and frameworks, each of which is applied to a problem in specific. In this work, the concept of quality of use can be highlighted, which is aimed at the ability to perform an activity and also the level of ease that users need to achieve a goal efficiently and with satisfaction [65].

This concept is directly linked to usability, which seeks to evaluate the quality of systems in different aspects, such as ease of learning and use, efficiency of use and productivity, user satisfaction, utility, the safety of use, among others defined by the designer as required (as seen in Section 3.4). Thus, when it comes to the usability of a system, the objective is to determine if the user experience when using the product will be pleasant [13] and also, if the product works and performs the expected tasks and correctly. To achieve the goal of usability, it is necessary to understand the characteristics that define the term [54]. Usability is defined through five dimensions that an interface must include: being efficient, effective, tolerant of errors, attractive, and easy to learn. When all these dimensions are foreseen, it can be concluded that the interface reached the usability criteria, but for

this, it is necessary that usability evaluations are carried out, from the beginning of the software/language construction process [10], [79].

From these usability goals, the designer can, through questions, evaluate these different aspects that an interactive product must offer the user, as well as the experience that this user had when using a certain system. Thus, it should be understood that these usability goals, extracted through questions, are also transformed into usability criteria, for example, ability to learn to use the system, number of mistakes made in performing a task, among others. In order for us to make use of these questions and goals, we need to understand which evaluation methods, techniques or frameworks can be used to evaluate these systems. Among these, for this research, we highlight the Heuristic Evaluation and the Cognitive Dimensions Framework that was used in the development of the Heuristic Checklist and Usability Questionnaire that compose the Usa-DSL Process.

2.3 Heuristic Evaluation

Heuristics are based on continuous experience obtained through practical knowledge. Heuristic knowledge comes from a practice build-up over years of understanding "what works" and "what doesn't work" for a specific context of use [53].

Heuristic Evaluation (HE) is an **inspection method**. An inspection method does not involve end users. The analysis is performed by experts who advocate for the user, that is, they know what the users want and need, and know the possible HCI techniques evaluate whether a particular computational artifact provides a good experience for the user [76].

The purpose of a HE is to **identify problems** (for a particular user profile and task set), consisting of a group of three (3) to five (5) HCI experts who inspect the interface or, in our case, the DSL, without involving users using a heuristic list (guidelines), empirical basis with the intention of generating as a result a report of potential problems and recommendations for the evaluated solution [54].

The Heuristic Evaluation Method is designed to find usability issues during an iterative design process. It is a simple, fast, and inexpensive method to evaluate HCI when compared to empirical methods. The method is based on a set of usability heuristics, which describe desirable interaction and interface features.

The **heuristic evaluation method** was proposed by Jakob Nielsen in 1994 [53]. This is a heuristic-driven inspection - general principles of good interface design, aimed at maximizing artifact usability. Traditionally, ten (10) heuristics have been used, which have been altered and expanded since their original proposal, to cover new technologies and computing environments. In order to contextualize each of the following Heuristics, we present a brief description of them (extracted from [55]):

- **H1 - Visibility of system status:** "The design should always keep users informed about what is going on, through appropriate feedback within a reasonable amount of time."
- **H2 - Match between system and the real world:** "The design should speak the users' language. Use words, phrases, and concepts familiar to the user, rather than internal jargon. Follow real-world conventions, making information appear in a natural and logical order."
- **H3 - User control and freedom users:** "Users often perform actions by mistake. They need a clearly marked "emergency exit" to leave the unwanted action without having to go through an extended process."
- **H4 - Consistency and standards:** "Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform and industry conventions."
- **H5 - Error prevention:** "Good error messages are important, but the best designs carefully prevent problems from occurring in the first place. Either eliminate error-prone conditions, or check for them and present users with a confirmation option before they commit to the action."
- **H6 - Recognition rather than recall:** "Minimize the user's memory load by making elements, actions, and options visible. The user should not have to remember information from one part of the interface to another. Information required to use the design (*e.g.* field labels or menu items) should be visible or easily retrievable when needed."
- **H7 - Flexibility and efficiency of use:** "Shortcuts — hidden from novice users — may speed up the interaction for the expert user such that the design can cater to both inexperienced and experienced users. Allow users to tailor frequent actions."
- **H8 - Aesthetic and minimalist design:** "Interfaces should not contain information which is irrelevant or rarely needed. Every extra unit of information in an interface competes with the relevant units of information and diminishes their relative visibility."
- **H9 - Help users recognize, diagnose, and recover from errors:** "Error messages should be expressed in plain language (no error codes), precisely indicate the problem, and constructively suggest a solution."
- **H10 - Help and documentation:** "It is best if the system does not need any additional explanation. However, it may be necessary to provide documentation to help users understand how to complete their tasks."

2.4 Cognitive Dimensions

Originally the Cognitive Dimensions (CD) Framework was proposed as a tool for broad discussion of usability, as it offers a vocabulary that discusses the trade-offs that occur when designing programming environments and information artifacts, especially programming languages and tools [32] [31] [30]. At first, when developing the CD, the authors expected that the assessment would be carried out by someone who had knowledge about the structure and still had a good understanding of the system. For a while it was thought that users could be designers, although it is known that designers generally have design priorities over evaluation; or who was an HCI specialist, as in most HCI evaluations.

Thus, as the main objective was for end users to use the framework, Green and Blackwell [30] sought to enrich their conceptual vocabulary so that end users could use their concepts. By focusing on achieving the goal of end users using the framework extensively, the authors faced another restriction on its dissemination. They understood that advertising only to designers and HCI people would hardly get other people to use it. It was then that Kadoda *et al.* [39] took a new step towards the goal of expanding the use of the CD and introduced a Cognitive Dimensions questionnaire to carry out a usability assessment. This questionnaire uses the concepts of the Cognitive Dimensions Framework and aims to carry out the assessment by system users (end users), rather than by designers or specialist HCI. When analyzing and applying the questionnaire, the authors indicated that it was a good idea, as it not only motivates users to understand and use, but involves users throughout the work.

In the same perspective as the Kadoda questionnaire *et al.* [39], our study led us to use the Cognitive Dimensions Questionnaire [16] [17] as a basis for adapting our questionnaire. While the CD Framework and, consequently, the questionnaire assess important points of the system, such as encapsulation and information hiding, it does not consider representational issues, such as effectiveness (button size) and aesthetics, as it is based only on structural properties. For this reason, we understand that these questions can be added to the adapted questionnaire. In addition, we can suggest that the evaluator perform an additional heuristic evaluation, which is part of another evaluation flow of the Usa-DSL Process, to obtain other information about its language. For a better understanding of the reader, we have made available two versions of the questionnaire, that is, the Cognitive Dimensions Questionnaire and the Usability Questionnaire adapted in Appendix E.

In order to contextualize each of the following Cognitive Dimensions, we present a brief description of them (extracted from [16] [30]):

- **Viscosity** - “A viscous system needs many user actions to achieve a goal.” If the user needs to perform multiple actions to complete a task, it will result in system abandonment, leading to resistance to change. As an example, changing all references

equal without a search command might require one action per reference. "(Environments containing adequate abstractions can reduce viscosity.)"

- **Visibility** - ability to view components easily - "Systems that bury information in encapsulations reduce visibility. Since examples are important for problem-solving, such systems are to be deprecated for exploratory activities; likewise, if the consistency of transcription is to be maintained, high visibility may be needed."
- **Premature commitment** - constraints on the order of doing things - "Self-explanatory. Examples: being forced to declare identifiers too soon; choosing a search path down a decision tree; having to select your cutlery before you choose your food."
- **Hidden dependencies** - important links between entities are not visible - "If one entity cites another entity, which in turn cites a third, changing the value of the third entity may have unexpected repercussions. Examples: cells of spreadsheets; style definitions in Word; complex class hierarchies; HTML links. There are sometimes actions that cause dependencies to get frozen *e.g.* soft figure numbering can be frozen when changing platforms; these interactions with changes over time are still problematic in the framework."
- **Role-expressiveness** - the purpose of an entity is readily inferred - "Role-expressive notations make it easy to discover why the author has built the structure in a particular way; in other notations each entity looks much the same and discovering their relationships is difficult. Assessing role-expressiveness requires a reasonable conjecture about cognitive representations."
- **Error-proneness** - the notation invites mistakes and the system gives little protection - "Enough is known about the cognitive psychology of slips and errors to predict that certain notations will invite them. Prevention (*e.g.* check digits, declarations of identifiers, etc) can redeem the problem."
- **Abstraction** - types and availability of abstraction mechanisms - "Abstractions (redefinitions) change the underlying notation. Macros, data structures, global find-and-replace commands, quick-dial telephone codes, and word-processor styles are all abstractions. Some are persistent, some are transient. Abstractions, if the user is allowed to modify them, always require an abstraction manager – a redefinition sub-device. It will sometimes have its notation and environment (*e.g.* notepad) but not always (for example, a class hierarchy can be built in a conventional text editor). Systems that allow many abstractions are potentially difficult to learn."
- **Secondary notation** - extra information in means other than formal syntax - "Users often need to record things that have not been anticipated by the notation designer."

Rather than anticipating every possible user requirement, many systems support secondary notations that can be used however the user likes. One example comments in a programming language, another is the use of colors or format choices to indicate information additional to the content of the text."

- **Closeness of mapping** - closeness of representation to domain - We need to identify and design the notation as close to the user's real world. "How closely related is the notation to the result you are describing?"
- **Consistency** - similar semantics are expressed in similar syntactic forms - "Users often infer the structure of information artifacts from patterns in notation. If similar information is obscured by presenting it in different ways, usability is compromised."
- **Diffuseness** - verbosity of language - Some notations can be annoyingly long-winded, or occupy too much valuable "real-estate" within a display area. Big icons and long words reduce the available working area.
- **Hard mental operations** - high demand on cognitive resources - "A notation can make things complex or difficult to work out in your head, by making inordinate demands on working memory, or requiring deeply nested goal structures."
- **Provisionality** - degree of commitment to actions or marks - Despite the existence of strict restrictions on the order of realizing certain actions (premature commitment), it can be useful to do things in a provisional way, such as design choices. Users could sketch or play "what if" games. Because some notation systems do not allow this type of action, the users cannot play or make incomplete markings (uncompromised).
- **Progressive evaluation** - work-to-date can be checked at any time - Evaluation is an important part of a design process, the use of notational systems can facilitate the evaluation during the process. Since users can check their work at any time during development. This possibility helps to verify the steps performed on the project and it is correct or needs improvement.

2.5 Usa-DSL Framework

This section describes a framework to evaluate DSL usability, called Usability Evaluation for Domain Specific Language Framework (Usa-DSL) [62]. Usa-DSL Framework was developed to help Language Engineers and HCI specialists to evaluate the usability of Domain-Specific Languages. This framework is composed of steps, phases, and activities that seek to assist in the definition, execution, analysis and presentation of the results of

the developed evaluation. The Usa-DSL structure and details about its phases, steps and activities will be presented in this section.

Usa-DSL Structure

The Usa-DSL Framework structure is based on the project life cycle process [82], which is composed of phases, steps and activities (see Figure 2.6). Basically, Usa-DSL is organized in phases, in which a set of steps have to be taken. For each step in a phase, there is one or none activity that has to be executed. Notice that some steps, in certain phases, have no activities, *e.g.* step “2 - Ethical and Legal Responsibilities” in phase Analysis has no activity, while this same step in phase Execution has activity “E2 - Introduce the Form and Collect Signatures of Subjects”.

There are four phases in the Usa-DSL Framework: **P**lanning, **E**xecution, **A**nalysis, and **R**eporting (PEAR phases). Each phase can be split into a set of the following steps: 1 - Evaluators Profiles; 2 - Ethical and Legal Responsibilities; 3 - Data Type; 4 - Empirical Study Method (SE); 5 - Evaluation Method (HCI); 6 - Metrics; 7 - Gathering Instruments; 8 - Evaluation Instructions; 9 - Evaluation Conduction; 10 - Data Packaging and; 11 - Evaluation Reporting. Important to notice that the PEAR phases have to be executed, for each step, in that order. Finally, there are 32 activities that are distributed between phases and steps.

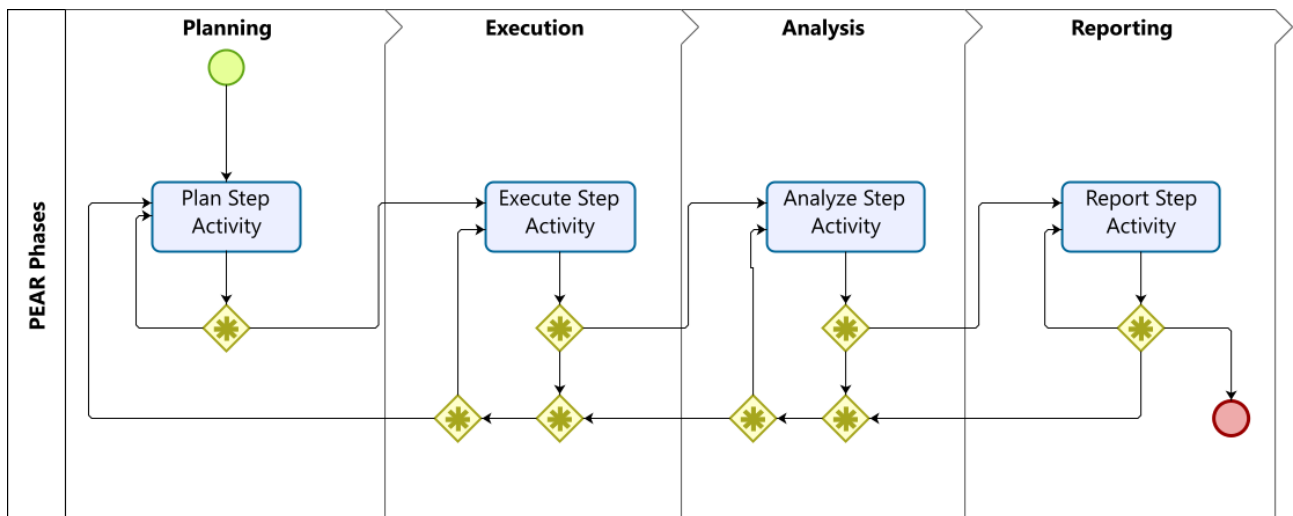


Figure 2.5: PEAR phases steps/activities order (BPMN notation)

The Usa-DSL Framework structure was planned in order to be adapted to the needs of each evaluation. It is possible to begin the “Planning” phase from any of the steps present in the Usa-DSL Framework. For example, the evaluator can start the evaluation planning by the “P1 Define Evaluators Profiles” activity, or by the “P3 Define Data Type” activity. This will improve the framework flexibility since it allows different evaluators to start the evaluation based on the activities that they feel more comfortable with, the ones that they already have some data, or even the activities that are easier to perform for a specific DSL. Besides, if the evaluator wants to perform a step in each of the PEAR phases, that also is possible, for example, it is possible to execute all activities from step “1 - Evaluators Profile” in all

PEAR phases before starting activities in any other step. Furthermore, not all steps have to be performed. Some of them might not be executed, for example, the “ 4 - Empirical Study Method (SE)” step is only needed if the end user will be involved. Figure 2.5 shows a high-level diagram of the order in which steps/activities in the PEAR phases can be executed.





Steps	Phases			
	 Planning	 Execution	 Analysis	 Reporting
1- Evaluators Profiles	P1 Define Evaluators Profiles	E1 Apply Instruments to Identify Profiles	A1 Analyze Evaluator Profiles	R1 Report Evaluator Profiles
2- Ethical and Legal Responsibilities	P2 Define Informed Consent Term	E2 Introduce the Form and Collect Signatures of Subjects		R2 Report Subjects Number and the Form Used
3 - Data Type	P3 Define Data Type			
4 - Empirical Study Method (SE)	P4 Define Empirical Study Method	E4 Develop and Conduct Protocol	A4 Analyze the Developed Protocol	R4 Report the Developed Protocol
5 - Evaluation Method (HCI)	P5 Define Evaluation Usability Type	E5 Prepare the Evaluation		R5 Report Conduction Evaluation
6 - Metrics	P6 Define Metrics for Language Validation			
7 - Gathering Instruments	P7 Define the Instruments of Data Gathering	E7 Data Collection	A7 Analyze the Collected Data	R7 Report Data Analysis
8 - Evaluation Instructions	P8 Define the Instruments of Instruction and Training	E8 Introduce Instruments of Instruction and Conduct Training		R8 Report the Instruments
9 - Evaluation Conduction	P9 Define Execution Place	E9 Execution of Tasks and Evaluation Conduction	A9 Analyze the Performed Tasks	R9 Report Tasks Analysis
10 - Data Packaging	P10 Define Data Storage	E10 Store Data Obtained		
11 - Evaluation Reporting	P11 Define Study Reporting		A11 Analyze the Documentation	R11 Report the Results and Analyzed Information
	Activities			

Figure 2.6: Usa-DSL Framework

Usa-DSL Steps

The Usa-DSL Framework is composed of eleven (11) **Steps**. The steps of the Usa-DSL Framework are described next (see Figure 2.6).

Step 1 - Evaluators Profiles: in this step, the evaluator profile is defined, instruments to identify the evaluator is applied, the evaluator profile is analyzed and a report on is written ([1], [7], [22], [25], [28]).

Step 2 - Ethical and Legal Responsibilities: similarly to the DECIDE Framework, which is an evaluation guide [66] and based on Resolution No. 466/12 [49], Usa-DSL follows

the best practices of ethical and legal issues to protect the user data, dignity, anonymity, and well-being. Furthermore, it has to include some description to inform the users that they can stop the evaluation at any time they are not comfortable with some aspects of the evaluation process. At the end of this step, all the signed documents from the subjects are organized.

Step 3 - Data Type: in this step the type of data that will be used is defined, *i.e* the evaluator defines whether the collected data is quantitative, qualitative, or both. This will depend on the method that will be used, for example, usability testing uses quantitative, while user observation can use qualitative data. Basically, this step contains only one activity that is performed during the Planning phase.

Step 4 - Empirical Study Method (SE): the Empirical Study Method suggested for Usa-DSL is based on the Wohlin *et al.* [86] proposal, which can be a survey, a case study, or a controlled experiment. These methods can be defined based on, for example, the evaluator's profile (Step 1) or the data that will be collected (Step 3). The Empirical Study Method can be used with other evaluation methods, *e.g.* usability testing or heuristic evaluation. However, the restrictions and characteristics of every method must be always respected.

Step 5 - Evaluation Method (HCI): the evaluation methods defined Usa-DSL can be user observation evaluation, for example usability testing, or inspection evaluation, for example, heuristic evaluation. The user observation evaluation must be applied when the study intention is to obtain the end users opinion about the DSL usability aspects. The inspection evaluation aims to verify the relevance of the language on the usability specialist level.

Step 6 - Metrics: the metrics used on Usa-DSL were defined from an SLR mapping [70]. They are comprehension/learning, ease of use, effort/conclusion time, observed complexity, and efficiency. These metrics will guide the definition of the evaluation instruments questions to be applied during the evaluation. Similarly to Step 3, this step has only one activity performed during the Planning phase.

Step 7 - Gathering Instruments: the instruments were based on the studies of [66] and [75], *e.g.* heuristic checklist, ergonomic checklist, questionnaires, interview, use observation, or user action recording.

Step 8 - Evaluation Instructions: according to Wohlin *et al.* [86], the evaluation instructions can be composed of manual, instruments or tasks to be performed. These instruments must be distributed and used when executing an empirical method. They are used, for example, to clarify the participants of the evaluation on what will be evaluated and when the evaluation will take place.

Step 9 - Evaluation Conduction: this is the step in which the aspects defined on the previous steps applied. Therefore, it is necessary that the previous steps were executed and tested thoroughly, before involving the evaluation participants. Hence, a pilot test must be executed prior to the application of the evaluation to the actual participants. This will

guarantee that the evaluation is viable. Furthermore, it is also important to guarantee that the needed number of participants will be achieved, otherwise, the results may not be statistically relevant, if a quantitative evaluation is being performed.

Step 10 - Data Packaging: when the evaluation is finalized, the used material for training and collected data should be stored in a safe place with easy access in order to allow the study replication when necessary. This will allow future language evaluation and its comparison with the news collected data.

Step 11 - Evaluation Reporting: this report must follow the evaluation method that was chosen in step “5 - Evaluation Method (HCI)”. Each evaluation method provides a specific report with different fields that must be filled.

Usa-DSL Phases

There are four **Phases** in the Usa-DSL Framework: **P**lanning, **E**xecution, **A**nalysis, and **R**eporting (PEAR phases) (see Figure 2.5). Each phase has a set of activities that is related to a respective step.

Phase 1 - Planning: in this phase, the evaluator organizes the planning of the aspects that will be used in order to evaluate the DSL. In this phase, documents must be defined and created, as well as decision-making about the data that has to be collected or what kind of user will be part of the evaluation, for example. To summarize, this phase is where the structure and planning of the evaluation will be constructed.

Phase 2 - Execution: in this phase, the documents created are used, subjects are recruited, environments are created and the evaluation is performed, following the already defined protocol.

Phase 3 - Analysis: this phase aims to accomplish the analysis of the artifacts created in the Planning and Execution phases. In the Planning phase, this analysis is executed in order for the documents to be adapted and, therefore, the decisions about the evaluation execution can be made. In this phase, the analysis is focused on the collected data and tasks created.

Phase 4 - Reporting: in this phase, the evaluator will register the used protocol, the created artifacts and analyzed data.

Usa-DSL Activities

The Activities of the Usa-DSL Framework are composed by a set of theories and practices used in software evaluations by the Human-Computer Interaction and Software Engineering fields. The used concepts were identified from some authors from the those areas. Figure 2.6 shows all activities from the Usa-DSL Framework. The set is composed by 32 *Activities* distributed among the *Phases* and guided by the Usa-DSL *Steps*. The description of each phase, step and activity can be consulted in the project repository¹.

¹ [urlhttps://github.com/lldevana/Usa-DSL/wiki](https://github.com/lldevana/Usa-DSL/wiki)

2.6 Software Process

Software Engineering is concerned with the quality of its processes to ensure the construction of quality products. Thus, two conceptions of quality can be observed, one that assesses the quality of the process used to develop the products and another that assesses the quality of these products that will be delivered and used by end users [68]. In the area of software process improvement and evaluation, several capability/maturity models have been developed by the Software Engineering community, and their use is well established in practice. However, as generic models seek to cover a wide range of software products, services, processes, and technologies, a current trend can be observed in the development of customizations of these generic process models for different domains, applications or techniques [81].

In this context, software products are becoming more and more complex, making it necessary to organize their development and evaluation process. For this, there are several software processes that follow different approaches, such as agile processes, or approaches based on artifacts and activities. However, these processes do not always meet the individual needs of a project and therefore need to be adapted. To mitigate the adaptation process, a representation of the concepts of process models is needed, this representation is performed through metamodeling.

In addition to the adaptations that can be performed, there are software process metamodels intended for independent development, such metamodels provide information for the development of new software processes [81]. In Software Engineering, several metamodels of software processes and software processes based on metamodels can be found. However, not all of them are necessarily used for software process development, such as the Usa-DSL Process, which is intended for the usability assessment process for DSL, and is based on the Software and Systems Process Metamodel Engineering Metamodel (SPEM).

2.7 Software and Systems Process Engineering Metamodel

Software and Systems Process Engineering Metamodel (SPEM) [58] is a process engineering metamodel and a conceptual framework that makes it possible to provide the necessary concepts to model, document, present, manage, change and execute development methods and processes. The implementation of this metamodel is aimed at process engineers, project leaders, project managers, and developers who are responsible for maintaining and implementing individual processes or for their organizations.

SPEM is used to define software and systems development processes as well as their components. Its aim is to support a wide variety of development methods and

processes from different styles, cultural backgrounds, levels of formalism, life cycle models and communities. This metamodel enables the developer to choose the generic behavior modeling approach that best meets their needs. It even provides a specific framework to enhance those generic behavior models that are characteristic for describing different processes, focusing on providing the necessary additional information structures about processes modeled with UML or BPMN activities, for the purpose of describing a real process.

One of the main characteristics of the SPEM metamodel is the possibility of specifying different processes from a common knowledge base independent of the specific process. For this, a clear separation between Method Content and Process is defined. The first represents a knowledge base, while the second represents a lifecycle of a specific process. In other words, the knowledge base stores all the information necessary to be consumed in the process' life cycle. In subsection 2.7.1 the concept of Method Content and Process will be presented in detail, as well as the components they store, generate and consume.

2.7.1 SPEM Elements

The SPEM metamodel is composed of *Method Content* and *Process* as shown in Figure 2.7.

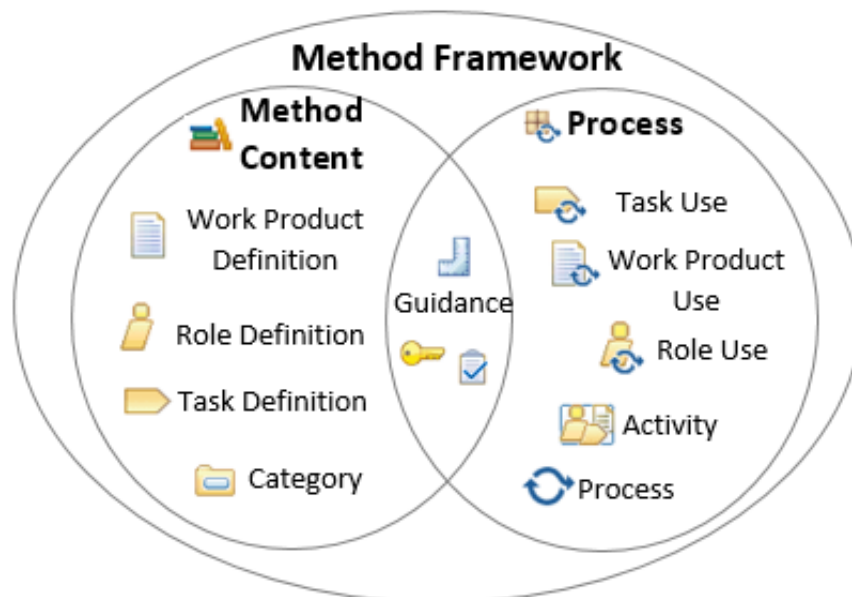


Figure 2.7: Method Framework SPEM

The Method Content describes what is to be produced, the skills needed, and a step-by-step explanation of how development goals will be achieved, regardless of where

these steps are placed within a development life cycle. The Method Content is composed of the elements: Work Product Definition, Role Definition, Definition, Category, and Guidance.

- Work Product Definition are work products/artifacts consumed, generated, or modified during the execution of a task. Roles use these work products to perform tasks and produce other work products as they run. They can also be used to define reusable assets, as they are part of the content of the defined method.
- Role Definition is an element of method content that defines a set of skills, competencies, and related responsibilities of an individual or a set of individuals, *i.e.* the roles performed within the process. Roles are used to define who execute each Task Definition, as well as to define those responsible for a set of Work Definitions.
- Task Definition describes a unit of work with a clear purpose, to be assigned to a specific Role Definition, in order to achieve a well-defined goal. This element provides complete explanations of how the task must be performed to achieve a given goal. This description is completely independent of when the task must be performed within the process lifecycle. Therefore, it does not describe when the work is done, but how and what work should be done during the lifecycle stages.
- Category is a flexible way to define different groupings for content categories. A Category is defined by Discipline, Role Sets, Domains, and Tools.
 - Discipline is a grouping of work, based on the similarity of concerns and cooperation of the work effort, that is, it is a collection of Tasks related to a main “area of concern” in the overall project
 - Role Set is a set of Roles organized by categories. This set of roles serves to group similar roles, for example, “Domain Analyst” and “Analyst” both perform similar techniques and have overlapping skills but are needed as separate roles for a method.
 - Domain is a logical grouping of work products that have affinity with each other based on resources, time, or relationship.
 - Tools is a container/aggregate for Tool Mentor that can provide general tool descriptions and capabilities.
- Guidance is a descriptive element that provides additional information related to Describable Elements. Specific Guidances are classified by types that have guidelines on the specific content structure or content that should be used. Guidance examples: Guidelines, Templates, Checklists, Example, Term Definition, Report.

Process defines sequences of how the work is being performed by functions/roles, as well as the work products being produced and evolved over time, that is, they describe

their life cycle. A process allows you to express the Task Use, Role Use, Work Product Use, Activity, Process, and Guidance elements.

- Task Use is a method content and a work breakdown element that represents a Task, which is being performed by a Role in the context of a specific Activity. Each split structure can define different Task Use relationships for Work Product Use and Role Use. Therefore, a Task Definition can be represented by many Task Uses, but each within the context of an Activity with its own set of relationships.
- Role Use is a special division element that represents a performer of an Activity or a participant in the Activity. Role Uses are valid and specific only to the context of an Activity. They must not be reused in all process activities. Role Uses are profiles that are defined in the Role Definition, so they already have defined roles. At this stage of the process they will only be added for the Activity that should execute.
- Work Product Use represents a Work Product Definition in the context of a specific Activity. The Work Product Definition can be represented by many Work Product Uses, within the context of an Activity with its own set of relationships. They represent the products consumed, and generated during the execution of an activity.
- is a work breakdown element that defines basic units of work within a process as well as the process itself. This element is a concrete Work Product Definition that represents a general unit of work attributable to specific performers represented by a Role Use. An Activity can take input and produce output, it also represents a grouping element for other work breakdown elements such as Task Use, Role Use, and Work Product Use. An Activity can also represent an instance of another Activity, through the association used, which allows the second Activity to inherit its full substructure.
- Process is the stage of organizing the Method Content elements, that is, these elements are selected and related in partially ordered sequences in a personalized way for each specific types of projects. A Process allows expressing the Activity, Role Use, Task Use, Process, and Guidance elements representing the instantiation of the elements from the elements defined in the Method Content.
- Guidance is consumed by both Method Content and Process. They have the same characteristics and the same units for the entire Method Framework as specified above.

2.8 Business Process Model Notation

Business Process Modeling Notation was developed by the Business Process Management Initiative, initially called the Business Process Management Initiative (BPMI),

and this notation has gone through a number of revisions. The objective was to unify the way in which companies performed the modeling of their processes. This notation is now maintained by the Object Management Group (OMG), which released the Business Process Model Notation (BPMN), and changed the name of the method to business processes. Thus, a more detailed standard for business process modeling was created, using a richer set of symbols and notations for business process diagrams[59].

From a high-level perspective, BPMN is aimed at participants and other stakeholders in a business process to increase their understanding through a practical visual representation of its steps. Its main purpose is to provide a notation that is easily understood by business users, facilitating communication between the parties. This communication starts from the business, in which the analysts create the initial sketches of the processes, passing on to the technicians responsible for the technology that will execute these processes, and, finally, to the business people who will manage and monitor the processes. In this way, BPMN creates a standardized bridge to the gap between business process, design, and process execution [59].

The notation is also useful for making process improvements, documenting them, and helping to identify steps that can be automated. It is characterized by a graphical notation used in order to specify business processes through a diagram. Furthermore, it provides a standard notation that is understandable to business users, and at the same time has a semantic representation of a complex process that can be understood by technical users. BPMN became widely used in order to describe processes through business process diagrams. In addition, the notation focuses on being used directly by the stakeholders who are responsible for designing, managing and carrying out business processes. However, this notation while needing to be understandable enough also needs to allow BPMN diagrams to be translated into software process components. BPMN has an easy-to-use flowchart notation, independent of any particular deployment environment, its main purpose is to provide a notation that is easily understandable by all users.

Starting with the analysts who create the initial sketches of the processes, through the technicians responsible for the technology that will execute these processes, and finally to the business managers who will manage and monitor the processes. Thus, BPMN creates a standardized bridge to the gap between the business process, design and execution process. For these diagrams to be developed and used, a series of elements are necessary to create flows and representations of activities, tasks, execution profiles, among others. Next, the elements that will be used for the development of the process that is the subject of study of this thesis will be presented.

2.8.1 BPMN Elements

For a better understanding of Usa-DSL flows, we will describe a small set of notation categories so that the reader of a BPMN diagram can easily recognize the basic types of elements and understand the diagram. Within the basic categories of elements, additional variation and information can be added to support the requirements for complexity without dramatically changing the basic look and feel of the diagram. The five basic categories of elements are [59]:

- *Flow Objects*: are the main graphical elements to define the behavior of a Business Process. There are three Flow Objects: Events, Activities, and Gateways.
- *Data*: provide information about what Activities require to be performed and/or what they produce. Data is represented with four elements: Data Objects, Data Inputs, Data Outputs and Data Stores.
- *Connecting Objects*: this connecting the Flow Objects to each other or other information. There are four Connecting Objects: Sequence Flows, Message Flows, Associations and Data Associations.
- *Swimlanes*: is a graphical container for partitioning a set of activities from other activities. BPMN has two different types of Swimlanes "Pool" and "Lane".
- *Artifacts*: are used to provide additional information about the Process. There are two standardized Artifacts, but modelers or modeling tools are free to add as many Artifacts as necessary. The current set of Artifacts includes Group and Text Annotation.

Now will displays a list and figure (Figure 2.8) of the basic modeling elements that are depicted by the notation.

- *Sequence Flow*: A connecting object that shows the order in which activities are performed in a Process and is represented with a solid graphical line. Each Flow has only one source and only one target. A Sequence Flow can cross the boundaries between Lanes of a Pool but cannot cross the boundaries of a Pool. - Normal flow refers to paths of Sequence Flow that do not start from an Intermediate Event attached to the boundary of an Activity.
- *End Event*: An Event that indicates where a path in the process will end. In terms of Sequence Flows, the End Event ends the flow of the Process, and thus, will not have any outgoing Sequence Flows. An End Event can have a specific Result that will appear as a marker within the center of the End Event shape. End Event Results are Message, Error, Compensation, Signal, Link, and Multiple.

- *Star Event*: As the name implies, the Start Event indicates where a particular Process will start.
- *Intermediate Event*: occur between a Start Event and an End Event. They will affect the flow of the Process, but will not start or (directly) terminate the Process. One use of Intermediate Events is to represent exception or compensation handling. This will be shown by placing the Intermediate Event on the boundary of a Task or Sub-Process (either collapsed or expanded). The Intermediate Event can be attached to any location of the Activity boundary and the outgoing Sequence Flows can flow in any direction. However, in the interest of clarity of the Diagram, we RECOMMEND that the modeler choose a consistent location on the boundary. Type Dimension (e.g., None, Message, Timer, Error, Cancel, Compensation, Conditional, Link, Signal, Multiple, Terminate.)
- *Gateway*: A Gateway is used to control the divergence and convergence of Sequence Flows in a Process. Thus, it will determine branching, forking, merging, and joining of paths. Internal markers will indicate the type of behavior control.
- *Gateway Control Types*: Icons within the diamond shape of the Gateway will indicate the type of flow control behavior. The types of control include:
 - *Exclusive Gateway* decision and merging. Both Exclusive and Event Base perform exclusive decisions and merging Exclusive can be shown with or without the “X” marker.
 - *Inclusive Gateway* decision and merging.
 - *Complex Gateway* - complex conditions and situations (e.g., 3 out of 5).
 - *Parallel Gateway* forking and joining. Each type of control affects both the incoming and outgoing flow.
- *Message*: An Object that depicts the contents of a communication between two Participants. A message is transmitted through a Message Flow and has an identity that can be used for alternative branching of a Process through the Event-Based Exclusive Gateway.
- *Pool*: A pool represents a Participant in a Collaboration. Graphically, a Pool is a container for partitioning a Process from other Pools/Participants. A Pool is not required to contain a Process, i.e., it can be a “black box.” Pool but cannot cross the boundaries of a Pool.
- *Lane*: A partition that is used to organize and categorize activities within a Pool. A Lane extends the entire length of the Pool either vertically or horizontally. Lanes are often used for such things as internal roles (e.g., Manager, Associate), systems (e.g., an enterprise application), or an internal department (e.g., shipping, finance).

- **Sub-Process:** A Process that is included within another Process. The Sub-Process can be in a collapsed view that hides its details. A Sub-Process can be in an expanded view that shows its details within the view of the Process that it is contained in. A Sub-Process shares the same shape as the Task, which is a rectangle that has rounded corners.
- **Task:** is an atomic Activity that is included within a Process. A Task is used when the work in the Process is not broken down to a finer level of Process detail.
- **Text Annotations:** are a mechanism for a modeler to provide additional text information for the reader of a BPMN Diagram.
- **Activity:** Work that a company or organization performs using business processes. An activity can be atomic or non-atomic (compound). The types of activities that are a part of a Process Model are Process, Sub-Process, and Task.

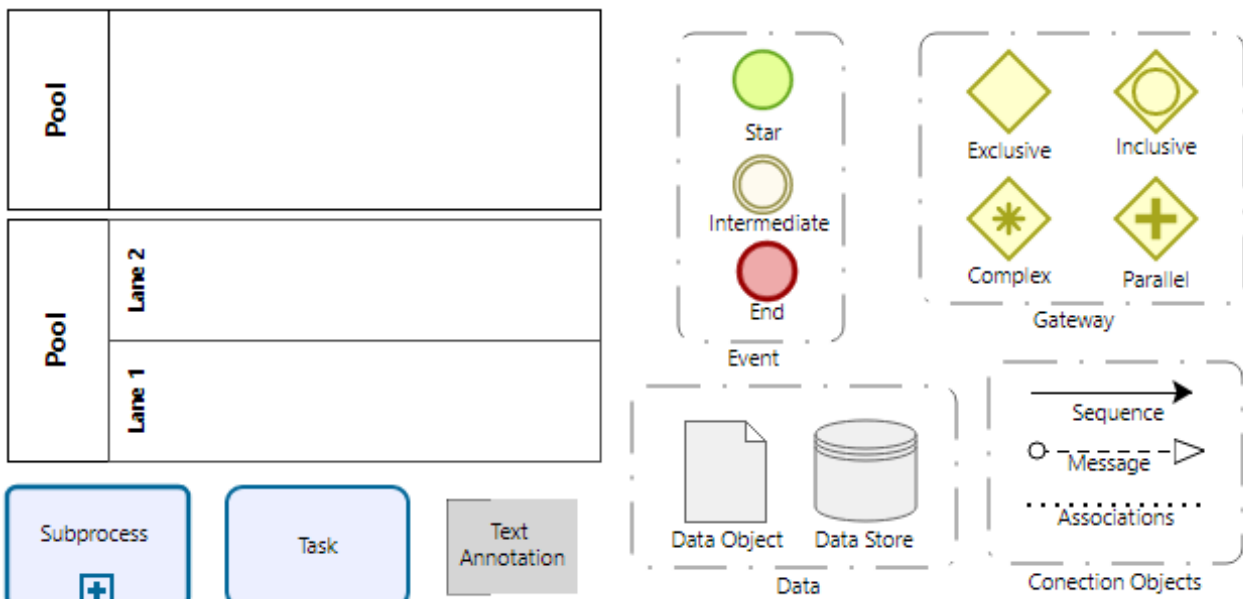


Figure 2.8: BPMN Elements

2.9 Final Considerations

In this chapter, the concepts that will be used throughout this work were presented. In a nutshell:

- The concept of Domain-Specific Language (DSL) was used to define the domain to which the process is applied to, *i.e.*, usability evaluation for DSL.

- Usability Engineering presented the definition of the application of evaluation methods, techniques, and procedures in the area of Human-Computer Iteration and guided the use of usability evaluation within this context.
- The Heuristic Evaluation defined the concept of usability inspection, and guided us in the development of the Heuristic Checklist for DSL evaluation.
- The concept of Cognitive Dimensions was used to support the adaptation of the Usability Questionnaire.
- The Usa-DSL Framework was the main concept used, as the Usa-DSL Process emerged from the need to generate usage flow, define activities, roles and artifacts.
- Software Process helped us to understand the definition of quality and process improvement, within the Software Engineering field, complementing the knowledge acquired with Usability Engineering that deals with the evaluation process that meets process quality.
- Software and Systems Process Engineering Metamodel (SPEM) and Business Process Model Notation (BPMN) provided us with the necessary support to develop the Usa-DSL Process, define its Principles, Profiles, Work Products, Tasks, Activities, Concepts, Life Cycle, Process, SubProcess, Flow among others.

3. SYSTEMATIC LITERATURE REVIEW

This chapter presents a Systematic Literature Review (SLR) to identify and analyze the trending topics on evaluation of DSLs [61]. We based our SLR on the Kitchenham protocol [40] and used the Thoth tool [46] to help in the gathering of information. This SLR replicates the protocol defined in a previous SLR [70]. The period in which the SLR was executed was from June 2016 to September 2020. This study allowed us to identify primary studies in both Human-Computer Interaction (HCI) and Software Engineering (SE) fields related to the thesis focus.

3.1 SLR Planning

In the planning stage, we performed the following activities in order to establish an SLR protocol: the establishment of the research goals and research question, definition of the search strategy, selection of primary studies, quality assessment, definition of the data extraction strategy, and selection of synthesis methods.

The goals of the study were to examine whether HCI aspects were considered or not during the development of a DSL, to know the techniques and approaches used to evaluate DSLs, and whether there were problems and limitations regarding DSL evaluation when HCI techniques were used. To achieve those goals we set the following three research questions:

- *RQ1*. Was the importance of usability considered during DSL development?
- *RQ2*. What were the evaluation techniques that were applied in the context of DSLs?
- *RQ3*. What were the problems or limitations identified during the DSL usage?

Search Strategy: the following digital libraries were used: ACM¹; IEEE²; ScienceDirect³; and Scopus⁴.

Selection Criteria: the following inclusion (IC) and exclusion (EC) criteria were used:

- *IC1*: The study must contain at least one of the terms related to HCI evaluation in DSLs in the title or abstract;
- *IC2*: The study must present some type of DSL evaluation;
- *EC1*: The study is about evaluation but not DSLs;

¹ACM: <http://portal.acm.org/>

²IEEE: <http://ieeexplore.ieee.org/>

³ScienceDirect: <http://www.sciencedirect.com/>

⁴Scopus: <https://www.scopus.com/>

- *EC2*: The study is not written in English;
- *EC3*: The study is about DSL but does not present an evaluation;
- *EC4*: The study is about HCI but not DSL evaluation.

3.2 SLR Execution

During this phase, the search string construction, studies selection, quality assessment, data extraction and synthesis were performed. The information produced during the execution of the SRL can be accessed at Zenodo⁵.

Search String Construction: the search string was built based on terms from DSL and HCI, from usage evaluation and usability, and their synonyms. Figure 3.1 shows the generic and specific string used in the digital libraries Table 3.1.

Figure 3.1: Search String

```
((“domain specific language” OR dsl OR dsm OR “domain specific modeling”
  OR, “domain specific modeling language” OR dsml) AND (evaluation OR
  evaluating OR experiment) AND (usability OR “user centered design” OR
  “user experience” OR hci OR “human computer interaction”))
```

Table 3.1: Specific String

Base	String
ACM	[[Publication Title: "Domain Specific Language"] OR [Publication Title: dsl] OR [Publication Title: dsm] OR [Publication Title: "Domain Specific Modeling"] OR [Publication Title: "Domain Specific Modeling Language"] OR [Publication Title: dsml]] AND [[Full Text: evaluation] OR [Full Text: evaluating] OR [Full Text: experiment]] AND [[Full Text: usability] OR [Full Text: "user centered design"] OR [Full Text: "user experience"] OR [Full Text: hci] OR [Full Text: "human computer interaction"]]
SCOPUS	((“Domain Specific Language” OR dsl OR dsm OR “Domain Specific Modeling” OR “Domain Specific Modeling Language” OR dsml) AND (evaluation OR evaluating OR experiment) AND (usability OR “user centered design” OR “user experience” OR hci OR “human computer interaction”))
SCIENCE DIRECT	("Domain Specific Language" OR dsl OR dsm OR "Domain Specific Modeling" OR "Domain Specific Modeling Language" OR DSML) AND (evaluation OR evaluating OR experiment) AND ("usability" OR "user centered design" OR "user experience" OR hci OR "human computer interaction")
IEEE	((“All Metadata”:“Domain Specific Language” OR DSL OR dsm OR “Domain Specific Modeling” OR “Domain Specific Modeling Language” OR dsml) AND (“All Metadata”:evaluation OR evaluating OR experiment) AND (“All Metadata”:usability OR “user centered design” OR “user experience” OR hci OR “human computer interaction”))

⁵DOI 10.5281/zenodo.4563198

Quality Assessment: This step was performed by two evaluators who analyzed each one of the studies and answered the quality assessment questions as follows: yes, partially, and no. Each answer was graded as: 1 for yes, 0.5 for partially, and 0 for no. After answering the 5 quality assessment questions, only studies that were scored 2.5 to 5 were considered for further analysis. Table 3.2 shows only the articles that were considered to be read. The final quality score is the average from the assessment of the two evaluator. The quality assessment questions were:

- QA1. Did the article make any contribution to HCI?
- QA2. Did the article present any usability evaluation technique?
- QA3. Did the article present the analysis of the results?
- QA4. Did the article describe the evaluated DSL?
- QA5. Did the article describe the encountered usability problems?

Table 3.2: Quality Assessment

Studies		Evaluator 1					Evaluator 2					Quality Score
Reference	Year	QA1	QA2	QA3	QA4	QA5	QA1	QA2	QA3	QA4	QA5	
[2]	2018	Y	Y	Y	Y	Y	N	P	Y	N	P	3.5
[4]	2017	P	N	Y	Y	N	N	Y	Y	N	Y	2.8
[12]	2018	Y	Y	Y	Y	P	P	Y	Y	Y	Y	4.5
[11]	2018	P	N	Y	Y	P	N	Y	Y	Y	Y	3.5
[19]	2019	P	N	Y	Y	P	N	N	Y	Y	N	2.5
[29]	2018	P	N	Y	Y	N	N	P	Y	P	P	2.5
[33]	2018	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	5.0
[35]	2019	Y	Y	Y	Y	Y	Y	P	Y	Y	P	4.5
[37]	2019	Y	Y	Y	P	P	P	P	N	Y	N	2.8
[41]	2018	N	Y	Y	Y	P	N	Y	Y	P	Y	3.5
[43]	2019	N	N	Y	Y	N	N	Y	Y	Y	Y	3.0
[44]	2018	P	N	Y	P	Y	P	N	Y	P	Y	3.0
[50]	2020	Y	Y	P	P	Y	Y	Y	Y	Y	Y	4.5
[51]	2019	P	N	Y	Y	N	N	P	Y	Y	P	2.8
[52]	2019	N	N	Y	Y	Y	P	Y	Y	Y	Y	3.8
[57]	2017	P	N	Y	Y	P	P	N	Y	Y	P	3.0
[63]	2018	Y	Y	Y	N	Y	Y	Y	N	N	N	3.0
[62]	2018	Y	Y	Y	N	N	Y	Y	Y	N	Y	3.5
[72]	2019	Y	Y	Y	Y	Y	N	Y	Y	Y	N	4.0
[77]	2019	Y	N	Y	Y	Y	N	N	Y	N	Y	3.0
[78]	2018	P	N	Y	Y	P	N	Y	N	Y	N	2.5

Score: (P) Partially = 0.5, (Y) Yes= 1; (N) No = 0

Primary Studies Selection: the performed search, based on the search string for each database, returned the number of studies presented in Figure 3.2.

In the first phase of the SLR, 44 papers returned from ACM, 39 from IEEE, 146 from Scopus, and 14 from ScienceDirect, resulting in 243 papers. When eliminating duplicate papers and applying the inclusion, exclusion, and quality criteria, 21 papers remained, which



Figure 3.2: Studies Selection Process

were thoroughly read. Figure 3.2 shows the number of papers that were selected after each phase.

3.3 SLR Analysis and Answers to Research Questions

This section presents the answers to the research questions from Section 3.1. The answers are based on the 21 studies selected in Section 3.2.

RQ1. Was the importance of usability considered during DSL development?

The importance of usability in the development or evaluation of DSL was discussed in most of the selected studies. However, some studies quote usability evaluation instruments but do not describe the process itself. In a nutshell: i) some of these studies evaluate the environment in which the DSL is developed, without evaluating the language itself; ii) some define usability criteria to evaluate the language, but without relating them to the quality of use criterion; iii) some studies compared GPL with the DSL also without discussing the usability process, which could be involved in the development and evaluation of these languages. In the next paragraphs we summarize each of the studies analyzed in this paper.

First we mention papers that are related to usability of DSL somehow, but do not consider usability during the DSL development [2] [33] [35] [43] [44] [52] [57] [72].

Alhaag *et al.* [2] presented a user evaluation to identify language usefulness. The effectiveness and efficiency characteristics were measured based on the results of a task given to the participants and the time spent to complete the task. Furthermore, five other characteristics were evaluated: satisfaction, usefulness, ease of use, clarity, and attractiveness, through a subjective questionnaire created in accordance with the Common Industry Format (CIF) for usability test reports.

Nandra and Gorgan [52] adopted an evaluation processes to compare the use of GPL to a DSL, but there was no discussion on usability criteria. They made a comparison between Python and the WorDel DSL, using as criteria the average time in which the participants perform a certain task, code correctness, syntax errors, number of interactions in the editing area code (such as mouse clicks and key presses), and task execution precision.

Nosál *et al.* [57] addressed the user experience, without relating that to the usability criteria. This study presented an experiment with participants who had no programming knowledge, which they were organized into two groups to verify whether a customized IDE would facilitate the syntax comprehension of a programming language when compared to a standard IDE.

Finally, Henriques *et al.* [33] presented a DSL usability evaluation through SUS (System Usability Scale), which is a numerical usability evaluation scale with a focus on effectiveness, efficiency, and user satisfaction; Liu *et al.* [43] evaluated the web platform that runs a DSL, but not the language itself; and, Rodriguez-Gil [72] used an adaptation of UMUX (Usability Metric for User Experience), which is an adaptation of SUS.

Regarding usability analysis during the DSL development process, the following studies considered that in their research [4] [12] [11] [19] [29] [37] [41] [50] [51] [63] [62] [77] [78].

Shin-Shing [77] indicated that more studies evaluate usability in terms of effectiveness and efficiency than about other usability criteria. In that study, a measure of the feasibility of Model-Driven Architecture (MDA) techniques was also made in terms of effectiveness and efficiency. However, there was no description of the process for usability evaluation.

Cachero *et al.* [19] presented a performance evaluation between two DSL notations, one textual and one graphical. For their evaluation, a Cognitive Dimensions Framework (CDF) was used. This framework defines a set of constructions to extract values for different notations, so that at least, partially, the differences in language usability are observed. For example, the extent to which a product can be used by participants to achieve particular goals such as effectiveness, efficiency and satisfaction in a specific use context.

Hoffmann *et al.* [37] stated that, despite being a very important task, usability evaluation is often overlooked in the development of DSLs. For DSLs that are translated into other languages, a first impression of the efficiency of a DSL can be obtained by comparing the number of lines of code (LoC) to the generated output. In their paper, they presented a heuristic evaluation in which some cognitive dimensions of the CDF are observed. However, an evaluation is not performed.

Msoa [51] observed through a survey of studies that the Computer-Interpretable Guideline (CIG) has no emphasis on the usability of the modeling language, since aspects of usability or human factors are rarely evaluated. This can result in the implementation of inappropriate languages. Furthermore, incompatible domain abstractions between language users and Language Engineers remain a recurring problem with regard to language usability. To evaluate the presented language, a survey was conducted, using the System Usability Scale (SUS) questionnaire, in order to obtain the participants' perception of the DSL.

Le Moulec *et al.* [41] focused on the importance of DSL documentation, which, they claimed, imply in a better understanding of the language. In their study, a tool was

proposed to automate the production of documentation based on artifacts generated during the DSL implementation phase, and an experiment was carried out using the tool in two DSLs. Furthermore, they observed the efficiency in automating language documentation and, consequently, improving usability.

Barisic *et al.* [11] defined DSL usability as the degree to which a language can be used by specific users, to meet the needs of reaching specific goals with effectiveness, efficiency and satisfaction, within a specific context of use. The authors also mentioned that, although there is a lack of general guidelines and a properly defined process for conducting language usability evaluation, they are slowly being recognized as an important step. In another study, Barisic *et al.* [12] argued that there is still little consideration to user needs when developing a DSL. They also mentioned that, even though, the creation of DSLs may seem intuitive, it is necessary to have means to evaluate its impact. This can be performed using a real context of use, with users from the target domain.

Bacíková *et al.* [4] indicated that DSLs are directly related to the usability of the their domain. They also argued that it is important to consider domain-specific concepts, properties and relationships, especially when designing a DSL. The study also considered discussions related to the User Interface.

Poltronieri *et al.* [63] mentioned that domain engineers aim, through different languages, to mitigate difficulties encountered in the development of applications using traditional GPLs. One way to mitigate these difficulties would be through DSLs. Therefore, DSLs should have their usability adequately evaluated, in order to extract their full potential. In their study, a framework for evaluating the usability of DSLs was proposed.

Mosqueira-Rey and Alonso-Ríos [50] also indicated an increase in research related to the evaluation of usability of DSLs through subjective and empirical methods. Although there are studies based on heuristics for interface evaluation, there is a shortage with regard to heuristic methods.

In order to reflect the user needs, Gilson [29] found that usability related to Software Language Engineering (SLE) has been poorly addressed, despite DSLs directly involving end users. For the author, evaluating the theoretical and technical strength of a DSL structure is very common. Nonetheless, usability issues are often overlooked in these evaluations.

Table 3.3 presents a summary of the topics that were discussed in each analyzed paper regarding the importance of usability considered during the DSL development.

RQ2. What were the evaluation techniques that were applied in the context of DSLs?

Researchers use quantitative (14/21) or qualitative (13/21) data to analyze the DSL usability. Several use both (8/21) and only in two situations we could not identify the data type that was used by the researchers.

Table 3.3: Importance of Usability

Paper	RQ1	Main Topic regarding DSL Usability Analysis
Alhaag <i>et al.</i> [2]		Observes metrics of effectiveness, efficiency, satisfaction, usefulness, ease of use, clarity, and attractiveness.
Bacíková <i>et al.</i> [4]	✓	Presents a survey in which the authors claim that DSLs are directly related to domain usability.
Barisic <i>et al.</i> [12]	✓	Authors argue that it is necessary to have means to evaluate the impact of usability on DSL.
Barisic <i>et al.</i> [11]	✓	Usability is the degree to which a language can be used by specific users, to meet the needs of reaching specific goals with effectiveness, efficiency and satisfaction, within a context of specific use.
Cachero <i>et al.</i> [19]	✓	The authors apply the Cognitive Dimensions Framework(CDF) in their evaluation.
Gilson [29]	✓	The author mentions that evaluation of the theoretical and technical strength of the structure of a DSL is very common, however usability issues are often neglected in those evaluations.
Henriques <i>et al.</i> [33]		Does not consider or mention usability in the development of DSL.
Hesenius and Gruhn [35]		Does not consider or mention usability in the development of DSL.
Hoffmann <i>et al.</i> [37]	✓	Cite other studies that argue that the evaluation of usability is a very important task but often neglected in the development of DSL.
Le Moulec <i>et al.</i> [41]	✓	They emphasize the documentation of DSL, a factor that is considered essential in the understanding and use of languages, directly implying their usability.
Liu <i>et al.</i> [43]	✓	Presents a usability evaluation of the web platform that runs a DSL.
Logre and Déry Pinna [44]		Does not consider or mention usability in the development of DSL.
Mosqueira-Rey [50]	✓	Authors present a set of heuristics to evaluate usability of DSLs.
Msosa [51]	✓	The authors used the System Usability Scale (SUS) Questionnaire in their study to evaluate their DSL.
Nandra and Gorgan [52]		Does not consider or mention usability in the development of DSL.
Nosál <i>et al.</i> [57]		Analyzes the user experience when using a DSL.
Poltronieri <i>et al.</i> [63]	✓	Presents a framework to evaluate usability of DSLs.
Poltronieri <i>et al.</i> [62]	✓	Presents a framework to evaluate usability of DSLs.
Rodríguez-Gil [72]		They mention the concern to develop their DSL as close as possible to the domain user reality.
Shin-Shing [77]	✓	Presents a survey that somewhat evaluates this aspect.
Silva <i>et al.</i> [78]	✓	Presents a comparative evaluation between two languages in order to evaluate the usability of their DSL.

Legend: **RQ1** Was the importance of usability considered during the DSL development?

Regarding technique, Usability Evaluation (8/21) was the most used one. Other techniques were also used, *i.e.*, Usability Testing and Heuristic Evaluation. Several papers (8/21) did not describe which technique was used.

Different instruments were applied in the usability evaluation. Questionnaire was the most used instrument (13/21), but other instruments to support data gathering, such as logs, scripts, interviews, audio and video recordings and other tools that observe the

tasks performed by the users during the evaluation were also used. Only two papers did not describe the instruments that were used in the usability evaluation.

Table 3.4 shows a summary of the evaluation techniques, instruments and data types that each study uses.

Table 3.4: Evaluation Data

Paper	Data Type		Evaluation Techniques	Instruments
	Quantitative	Qualitative		
Alhaag <i>et al.</i> [2]	✓		Not described	Questionnaire
Bacikova <i>et al.</i> [4]		✓	Usability Evaluation	Task Recording
Barisic <i>et al.</i> [12]	✓	✓	Usability Evaluation	Questionnaire
Barisic <i>et al.</i> [11]	✓	✓	Usability Evaluation	Questionnaire; Video recordings
Cachero <i>et al.</i> [19]	✓		Not described	Manual Evaluation of Tasks
Gilson [29]	✓		Not described	Questionnaire
Henriques <i>et al.</i> [33]	✓		Usability Evaluation	Questionnaire
Hesenius and Gruhn [35]	✓	✓	Evaluates some HCI metrics (e.g. efficiency and effectiveness)	Recording of Sessions
Hoffman <i>et al.</i> [37]	Undefined	Undefined	Heuristic Evaluation	Undefined
Moulec <i>et al.</i> [41]	✓	✓	Not described	Questionnaire
Liu <i>et al.</i> [43]		✓	Usability Evaluation	Data Logs; Interviews
Logre and Déry Pinna [44]	✓	✓	Not described	Manual Method
Mosqueira-Rey and Alonso-Ríos [50]	Undefined	Undefined	Heuristic evaluation	Undefined
Msosa [51]	✓	✓	Usability Evaluation	Questionnaire
Nandra and Gorgan [52]	✓		Not described	Questionnaire; Recorded Actions
Nosal <i>et al.</i> [57]		✓	Not described	Task Recording
Poltronieri <i>et al.</i> [63]		✓	Usability Evaluation	Questionnaire
Poltronieri <i>et al.</i> [62]		✓	Usability Testing and Heuristic Evaluation	Questionnaire
Rodríguez-Gil <i>et al.</i> [72]	✓		Not well defined, observes efficiency and effectiveness metrics	Questionnaire (UMUX)
Shin [77]	✓	✓	Not described	Questionnaire
Silva <i>et al.</i> [78]	✓	✓	Usability Evaluation	Questionnaire (SUS)

RQ3. What were the problems or limitations identified during the DSL usage?

The analyzed papers present some limitations or problems regarding, either the evaluated DSL or the evaluation process that they performed. From the 21 selected articles, not all of them present the problems found in their DSL (see Table 3.5). Only Hesenius and Gruhn [35], Henriques *et al.* [33], Liu *et al.* [43] and Nosal *et al.* [57] somehow present the encountered problems. The other articles have limitations found in general, that is, in the designed DSL or in the evaluations. Bacíková *et al.* [4], Rodríguez *et al.* [72] and Silva *et al.* [78] did not present any problem or limitation in their studies.

Regarding the study by Hesenius and Gruhn [35], the GestureCards notation has some limitations in special circumstances, and two potential problems were described: being voluminous and in the description of spatial gestures. These problems occur because GestureCards uses spatial positioning to denote temporal relations of partial gestures. Thus, the volume problem occurs when the gestures are composed of multiple partial gestures that are defined separately. It was suggested that to avoid those problems, the parallel gestures

can be combined in a graphic representation when shared with the other features. The authors argue that this will be sufficient for most of the use cases.

Table 3.5: DSL Constraints and Evaluation Constraints

Reference	Description
DSL Constraints	
Henriques <i>et al.</i> [33]	Not all stakeholders participate in the evaluation.
Hesenius and Gruhn [35]	Notation introduced in this paper present problems in description of space gestures and bulky gestures.
Moulec <i>et al.</i> [41]	The use learning curve of the DSL is a challenge to participants.
Nosál <i>et al.</i> [57]	The syntactic noise still remains the main problem of pure EDSL when used by non-programmers.
Evaluation Constraints	
Alhaag <i>et al.</i> [2]	The evaluation was made by expert users and do not show the full potential of solution.
Barisic <i>et al.</i> [12]	The reliability of participants answers in the questionnaires, since they were young.
Cachero <i>et al.</i> [19]	According to the values observed in the cognitive dimensions (CDF), it can be inferred that these dimensions do not affect the same way that the maintenance of the domain model and the tasks of creating the domain model.
Mosqueira-Rey and Alonso-Ríos [50]	Some identifiers are not so clear and not so easy to remember.
Nandra and Gorgan [52]	GPL demonstrates better performance than DSL in syntax and time required for task description,
Poltronieri <i>et al.</i> [63]	Provide a Usability Framework. Does not provide a process that guides the user in decision-make.
Poltronieri <i>et al.</i> [62]	Does not provide a process that guides the user in decision-make.
Shin <i>et al.</i> [77]	Use of participants without professional experience in the sampling of the experiment.

Liu *et al.* [43] discussed the organizational problems for business people, as employees use their own mobile devices to process workflow tasks. Due to that, a middleware-based approach, called MUIT (Mobility, User Interactions and Tasks), was introduced to develop and deploy mobility, user interactions and tasks in Web Services Business Process Execution Language (WS-BPEL) mechanisms. This DSL allows to significantly reduce the manual efforts for developers with regard to user interactions, avoiding to use more than one type of code and thus offering satisfactory support for user experiences. On the one hand, the authors pointed out that some users from the healthcare area still complain that MUIT touch controls are not good when they process electronic patient records. On the other hand, no limitations were pointed out.

Henriques *et al.* [33] presented the OutSystems platform, a development environment composed of several DSLs, used to specify, build and validate data from Web applications on mobile devices. The DSL for Business Process Technology (BPT) process modeling, had a low adoption rate due to usability problems, increasing maintenance costs. Furthermore, they found a limitation related to population selection, because, due to resource

limitations, all participants in the usability experiments were members of OutSystems. The authors believe that business managers should also be invited and that more experiments should therefore be conducted with interested parties.

Mosqueira-Rey and Alonso-Ríos [50] also identified usability problems in their case study, which uses heuristics that help to identify real usability problems. As problems they pointed out that DSL identifiers do not have a clear meaning, some acronyms have no obvious meaning and certain identifiers are difficult to remember.

Nosal *et al.* [57] results indicate that even IDE customizations can significantly alleviate the problems caused by syntactic problems in the language. As limitations, the authors mentioned the low representativeness of Embedded DSLs (EDSL) when compared to real-world DSLs. An EDSL can be much more complex from a syntactic point of view, as it can include variables, functions, structures, etc. The benefits of the proposed technique (for example, file templates) may become insignificant due to the complexity of the language syntax. Therefore, generalization of the results for all EDSLs is not possible and a replication of the experiment with a more complex EDSL is necessary. In terms of domain abstractions, the study by Msosa [51] mentioned that the incompatibility of some domain abstractions is a limitation that can present some barriers and negatively impact the usability of a DSL.

Shin-Shing [77] analyzed techniques for reverse engineering and model transformations in Model Driven Architecture (MDA). Their evaluation was performed using usability metrics, *i.e.* productivity and efficiency. In their paper, they concluded that such techniques are still immature and superficial, requiring further studies in order to improve them. They emphasized that, before performing further usability evaluations, they first needed to pay more attention to the techniques and methods used during the MDA development phases.

Nandra and Gorgan [52] presented and evaluated the DSL WorDeL interface, which was designed to facilitate the connection of existing processing operations in high level algorithms. The authors compared the WorDeL DSL with the Python language. In their comparison, Python showed better results in terms of time to describe each task. Besides, the WorDeL DSL produces better results in terms of accuracy, with higher percentages of tasks completed correctly. WorDeL was also better in terms of accuracy when evaluated for its expressiveness.

Hoffmann *et al.* [37] did not present limitations on the evaluation, but mentioned that the number of lines of code can be considered a problem with regard to DSL usability. The authors mentioned that “For DSLs that are translated into other languages, the efficiency of a DSL can be obtained by comparing the number of lines of code between the source program and the generated output”. They also used the McCabe’s cyclomatic complexity [47] to analyze the language. Both lines of code and cyclomatic complexity are quantitative measures that can be applied to roughly estimate the effort to understand, to test and to maintain a program. Furthermore, the DSL was not used by end users.

Le Moulec *et al.* [41] found limitations regarding the contextualization of the documentation. Specifically, the model is customized to correspond to the DSL documentation concept. However, the provided examples had no context that were easily related to the original model. Another limitation was related to code compliance, the generated documentation did not help new users to make the models faster. Furthermore, some participants read all the available documentation before starting the exercises, while others did that when they started to use the editor. Therefore, the subjects spent around 30 minutes to perform the basic exercise, showing that the initial learning curve is a challenge to be faced in future works. Similarly, Le Moulec *et al.*, and Logre and Déry-Pinna [44], mentioned that the relevance of the participants' choice was not analyzed, making this a limitation.

Barisicic *et al.* [11] mentioned that the selection of participants might be a limitation in their study. They tried to mitigate the problem through tutors that would help the participants when they needed to answer the questions. This helped to guarantee the validity and integrity of the results. Likewise, the study by Alhaag *et al.* [2] also presented the background of the participants as a limitation for their study. As a solution to mitigate this threat, Alhaag *et al.* indicated that the platform should be evaluated by domain experts with little technical knowledge in order to better explore new metadata in their domain. Like the previous two studies, Gilson *et al.* [29] also described the lack of evaluation with end users as a limitation for their study.

Poltronieri *et al.* [62] [63] presented a framework focused on evaluating the usability of DSLs, called Usa-DSL framework. The authors pointed out the lack of a flow or process that would help the participants in the creation of the evaluation, guiding their steps, what should be done and at what time to perform a certain activity. Unlike the framework of Poltronieri *et al.*, Barisic *et al.* [12] focused only on evaluating usability with end users and through experimental studies.

3.4 Evolution of Usability Evaluation for DSL

This section summarizes the evolution on research of usability evaluation of DSL from Rodrigues *et al.* [70]. In the past 5 years there was a significant increase in published studies on usability evaluation of DSL. The previous Systematic Literature Review (SLR) was performed without limiting the initial year and looked for papers that were published until 2016. In that SLR, 12 papers were selected. The current SLR selected 21 papers that were published from 2016 to 2020. This showed an increase of 75% in the number of selected papers that somehow are related to usability evaluation of DSLs. Hence, **usability evaluation of DSL is still a trending topic**. Actually, it seems that **there is an increase on the research in this area**. Some of the analyzed papers are more straightforward to mention that [1] [12] [33] [37] [43] [63] [79] than others.

One interesting point from the previous SLR to this one, is that, in the previous SLR authors seemed to be interested to evaluate their DSL, without any concern on the protocol, technique and instruments that were used [1] [22] [28]. In the current SLR it was clear the increased concern on the development or organization of the techniques and instruments that are used to evaluate the DSL usability [5] [33] [35] [78]. Furthermore, new protocols, frameworks and even processes to evaluate the DSL usability have been proposed. For example, Barisic *et al.* [12] and Poltronieri *et al.* [63] developed frameworks to evaluate DSL usability focusing on the ease of use. On the one hand, Bacikova *et al.* [4] presented ways to automatize the evaluation, while Mosqueira-Rey *et al.* [50] have created some heuristics to evaluate DSL usability.

Although authors have increased their preoccupation to use well-defined techniques, instruments and processes to evaluate usability of DSLs, they are still neglecting to better describe the problems or limitations that they find when evaluating DSL usability. Usually, authors describe problems or limitations when the DSL is used, but neglect to describe the DSL usability problems. It seems that the focus has been on whether users can use the language rather than to identify usability problems that the language contains. For example, few studies present user perception, use satisfaction, system intuitiveness, real-world representation, among others. Some authors [35] [37] even use some terms (*i.e.* number of lines of code, usage time, efficiency and efficacy) that are not directly related to usability. This might be one of the reasons that several DSLs are not successful.

3.5 Evolution Taxonomy for DSL Evaluation

Based on the selected studies and the research questions, this section presents a taxonomy extension of terms used during the evaluation of DSLs. This taxonomy was structured as a conceptual mapping (see Figure 3.3). This taxonomy was based on the terms that were mentioned in the studies selected in our previous SLR [70] and in the SLR presented in this paper.

Figure 3.3 shows the main groups of categories represented as the external rectangles: framework/approaches, data type, usability evaluation methods, instruments, profile user, software engineering evaluation method and evaluation metrics. In each of these groups, there is a set of categories, for example, a user profile can have the following categories: HCI expert, DSL expert, potential user, final user. This figure also presents new categories and one new group (*i.e.* process) that were not present in the previous taxonomy. These new categories are represented by gray rectangles in the figure. Basically the new categories are: i) frameworks: Usa-DSL framework and USE-ME; ii) process: Usa-DSL process; iii) instruments: data log, challenge solution, video recording, task recording; iv) Software Engineering Evaluation Method: *quasi*-experiment; and, v) metrics: conciseness, readability

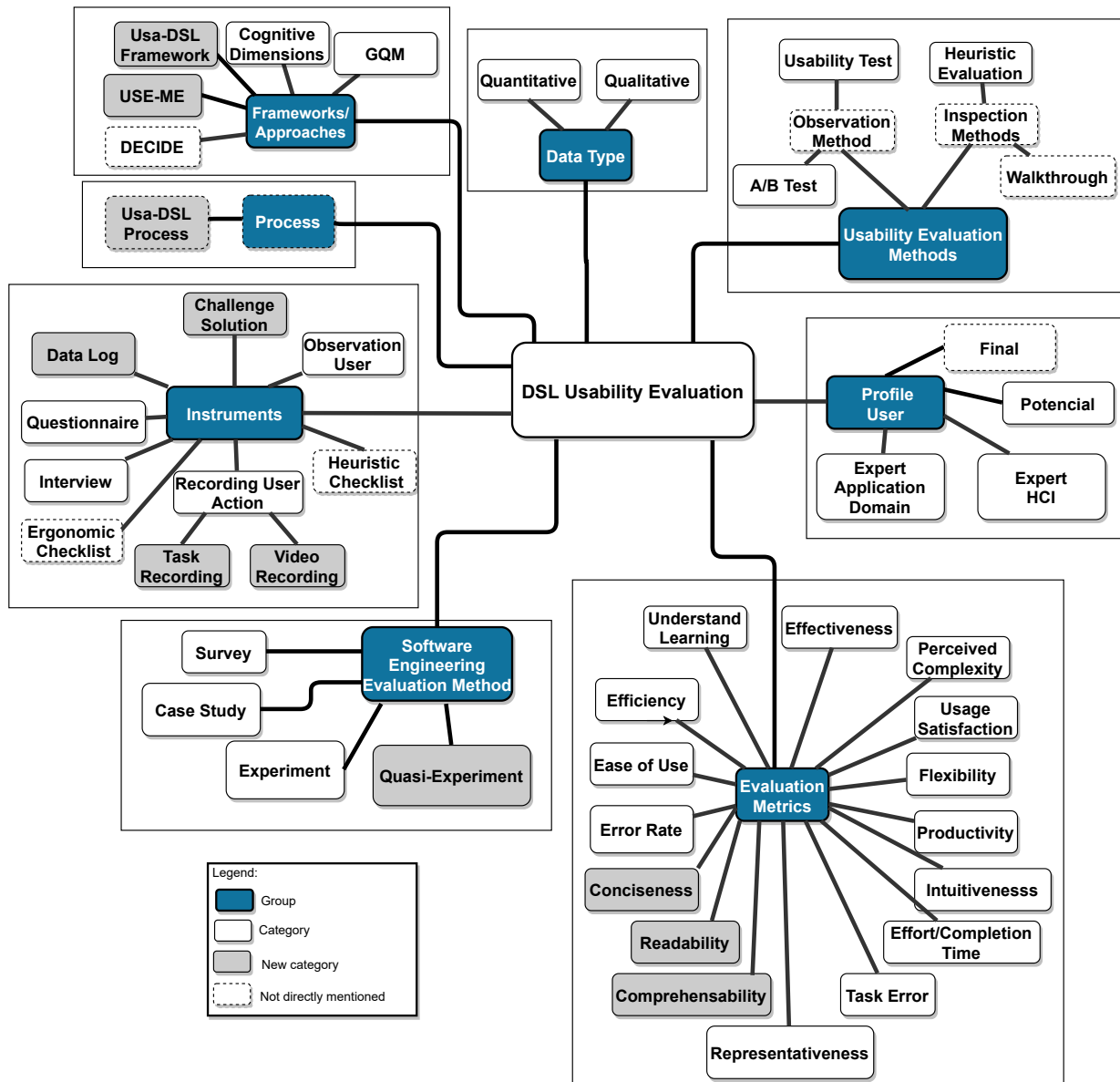


Figure 3.3: DSL Usability Evaluation Taxonomy

and comprehensibility. Furthermore, the figure also shows some categories represented by dashed round rectangles. These categories are not directly mentioned in the studies presented neither in the previous or in this SLR, but are important in the development of a framework and process to evaluate DSLs.

Similarly from what we found when answering the research questions presented in this paper, this taxonomy reflects the lack of standards when planning and applying DSL usability evaluation. We can highlight that many of the authors mention that they do not use standard techniques, methods or instruments when they evaluate their DSL. However, some authors have been developing new frameworks and processes to systematize and help Language Engineers to use standard methods, instruments and metrics for usability evaluation of DSLs.

Usa-DSL Metrics - Taxonomy

Based on the terms presented in the taxonomy, we conceptualize each of the metrics according to the definitions used by the authors mentioned in the SLR. These metrics will later compose the mapping between the cognitive dimensions presented previously. A brief description of the metrics are presented next:

- **Efficiency** - is the user capability to correctly implement a flow from a given high-level description of the required flow. Also Efficiency metrics are related to the time spent by the users and their productivity when performing the tasks [9], [75], [79];
- **Ease of Use** - ease of use is a basic concept that describes how easily users can use a product. Design teams define specific metrics per project — *e.g.*, simple to use, user friendly, flexible, requires the fewest steps possible to accomplish what I want to do, recovery from mistakes quickly and ease [8], [79];
- **Error Rate** - the error rate is defined by accounting for different types of errors made by the user, for example, performing a task different from what was requested, performing the task in half, among others. To account for the error rate of an application, a weighted sum of the errors made must be made, or captured by an application [83];
- **Task Error** - another measurement involves counting the number of errors the participant makes when attempting to complete a task. Errors can be unintended actions, slips, mistakes or omissions that a user makes while attempting a task. You should ideally assign a short description, a severity rating and classify each error under the respective category [9], [79];
- **Conciseness** - is the quality of being short and clear, and expressing what needs to be said without unnecessary words, that is, the users should not have to wonder whether different words, situations, or actions mean the same thing [41]. "DSL Conciseness, which refers to the economy of terms without harming the artifact comprehension" [1];
- **Readability** - is the ease with which a reader can understand written text. In natural language, the readability of the text depends on its content and presentation, for example, a researcher may look at the time required to read source code. The construct validity of this measure is the extent to which the readability of source code is actually related to the time required to read it [5];
- **Comprehensibility** - the quality of being easy or possible to understand. Comprehensibility refers to whether listeners can understand the message communicated by a speaker. "[...] some notations are less readable and harder to understand than others" [35];

- **Representativeness** - is used when making judgments about the probability of an event under uncertainty, and assist in the judgment or decision-making. "Representativeness dimension is associated to the conative function given that it works around the effect caused by the product in its potential user." Is characterized by the fact that it transmits a message in order to convince the user [1];
- **Intuitiveness** - "intuitive design is used informally to describe designs that are easy to use. So, when a user is able to understand and use a design immediately—that is, without consciously thinking about how to do it—we describe the design as “intuitive” [9];
- **Productivity** - "Productivity metrics are used to track and measure how efficient your team is in getting their tasks done. These metrics are used to manage and improve performance, as well as highlight where you need to improve" [8], [28];
- **Flexibility** - "degree to which a product or system can be used with effectiveness, efficiency, freedom from risk and satisfaction in contexts beyond those initially specified in the requirements" [1], [8], [28], [38] and [79];
- **Usage Satisfaction** - "degree to which user needs are satisfied when a product or system is used in a specified context of use" [1];
- **Perceived Complexity** - "the degree to which an innovation is difficult to understand and use" [73];
- **Understand Learning** - this metric is related to the understanding and learning of the notation and meaning of the program [8];
- **Effort/Completion Time** - efficiency is measured in terms of task time. that is, the time (in seconds and/or minutes) the participant takes to successfully complete a task. The time taken to complete a task can then be calculated by simply subtracting the start time from the end time as shown in the equation below: $\text{Task Time} = \text{End Time} - \text{Start Time}$, [22].

3.6 Final Considerations

Our SLR has shown that issues related to usability, user experience and design principles have been increasingly considered in the DSL project. Recognizing and understanding the relationship among these principles in the development and evaluation of DSL has been a challenge, as not all interaction design goals have the same relevance in all types of systems and, in the case of DSL, in the DSL domain.

Therefore, DSL usability evaluation still remains an important research topic [4] [12] [11] [50] [63] [62], since there are still several issues that need to be improved. For example, several authors that evaluate DSL usability do not use HCI methods and in several papers they do not describe the evaluation execution in a proper way [2] [19] [29] [41] [44] [52] [57] [77]. Some of the authors evaluate the usability using well-defined instruments, but do not use specific methods, while others use *ad-hoc* methods to do that [35] [72]. Sometimes this lack of proper evaluation leads to the DSL being unsuccessful, because when the users are unable to use the language in its full, or need to follow many steps and remember different paths, they end up giving up.

Another relevant point is that language developers still do not use available usability frameworks or processes. Nonetheless, some researchers have proposed new ways to evaluate DSL usability. In this paper we presented an extension to the previous published taxonomy for usability evaluation of DSLs. It is interesting to notice that after five years of the first published taxonomy, new categories were used by the researchers when evaluating their DSL.

Although the focus of our SLR was not to analyze tools to support the usability evaluation of DSLs, this is a subject that should be better investigated. This would allow researchers to understand how usability evaluation of DSLs is planned, executed, analyzed and reported by those tools.

Part II

Usa-DSL Process Conception

4. USA-DSL PROCESS

The Usa-DSL Process was modeled using the Software and Systems Process Engineering Metamodel (SPEM) [58] and the Business Process Model and Notation (BPMN) [59] and the Usa-DSL Framework [62]. The Usa-DSL Process was developed with the support of the EPF Composer tool [24]. This tool is used in the elaboration, customization, and publication of processes.

Usa-DSL Process is a Usability Evaluation Process for Domain-Specific Languages (DSL) that, over time, is expected to cover a broad set of needs of DSL designers with regard to usability evaluation of DSLs. The whole process can be found at <http://lesse.com.br/usa-dsl>.

Usa-DSL Process Fundamental Principles: The Usa-DSL Process was designed based on the following principles [80], as can be seen in Figure 4.1:

Supporting: to assist the performing of usability evaluation in a practical and efficient way; Supporting Analyst, Developers, and Testers, when it comes to evaluating the usability of DSLs is essential for them to be increasingly accepted by end users. Therefore, the construction of a process that assists in the development of these usability evaluations and brings a language closer to the field of Software Engineering approximated to the knowledge of usability testing and heuristic evaluation, are fundamental to the success of the developed DSL. Thus, an effective evaluation should align the interests of the project participants and the understanding of such evaluations.

Continuous Feedback: to assist DSL development through constant feedback, promoting continuous improvement of the language. The Usa-DSL Process seeks through an iterative cycle to adopt a systematic approach to obtain frequent usability feedback in the shortest time possible, to the DSL development group, to better monitor its impact [9] always thinking about the continuous improvement of language. For this, we seek the integration of two complementary processes, language development, and usability evaluation.

User-Centered Evaluation: through all the building phases, minimizing the DSL rework and disuse [55] “The User-Centered Design (UCD) process outlines the phases throughout a design and development life-cycle, all while focusing on gaining a deep understanding of who will be using the product” [84]. When you focus on the user and not the product, you learn what works for your users, as well as what does not work, what pleases, and what frustrates them [13].

Productivity: to make DSLs evaluation easier and productive, aiming to reach usability criteria and usage satisfaction [79]. The Usa-DSL Process is considered easier and more productive because its structure is designed for evaluators who do not know the HCI field and have the need to perform usability evaluations. From this concern was developed this process with a language closer to the Software Engineering field, thus seeking to make this activity more productive. For this reason, the Usa-DSL was developed using a user-centered

design (an explicit understanding of users, tasks, and environments), seeking to achieve the criteria of usability and satisfaction of use.

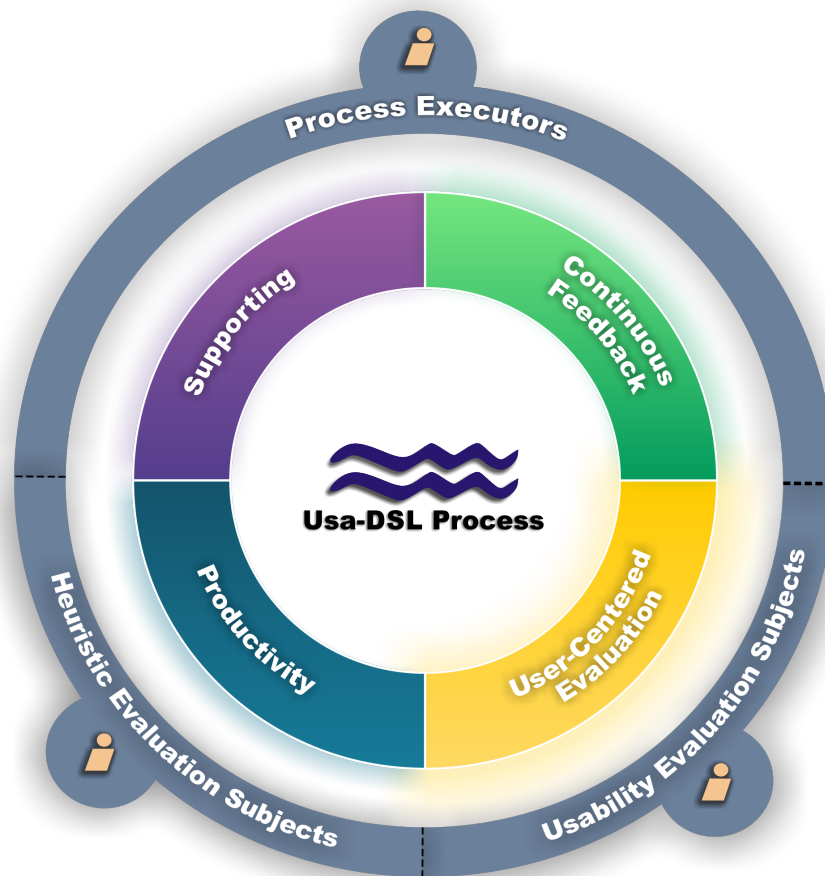


Figure 4.1: Usa-DSL Process Fundamental Principles

The Main Purpose of the Usa-DSL Process is to guide the question "Who will do what, when, and how?" [68], iteratively [82] and guide to the development of formal usability evaluations with easiness and speed.

The main objective is to provide the procedures to perform the evaluation regarding the usability of DSLs and verifying through methods, techniques, and activities how these languages meet the quality of use.

The Usa-DSL Process deals with the elaboration of usability evaluations involving the methods Heuristic Evaluation and Usability Testing [54], with the intention of systematizing and developing the evaluations in a productive way and with less cost.

The Usa-DSL Process has a structure composed of Method Content and Process [58], as can be seen in Figure 4.2. **Method Content** is a knowledge base, in which the information that will be consumed by the process is stored. The information stored in Method Content is organized into Work Products, Tasks, Profiles, Profile Set, and Steps. **Process** deals with the organization of the elements of the Method Content, relating these elements in sequence partially ordered and customized for each specific project. The process has basic

structural elements such as Phases, Activities, Profiles Use, Task Use, Use Work Products Use.

Usa-DSL contains a category of grouping elements that are intended for Guidelines, that is, templates for documents that will be consumed, generated, or modified throughout the life cycle. These templates will be used by both the method content and the process, they are described in a unique way.

The Usa-DSL Process has several artifacts of this type that can be used during the execution of a task and/or the process generated, are: Informed Consent Term, Profile Questionnaire, Usability Questionnaire, Heuristic Checklist, Glossary, Documentation, DSL Guide, Use Scenario, Training Documentation DSL, Study Protocol among others.

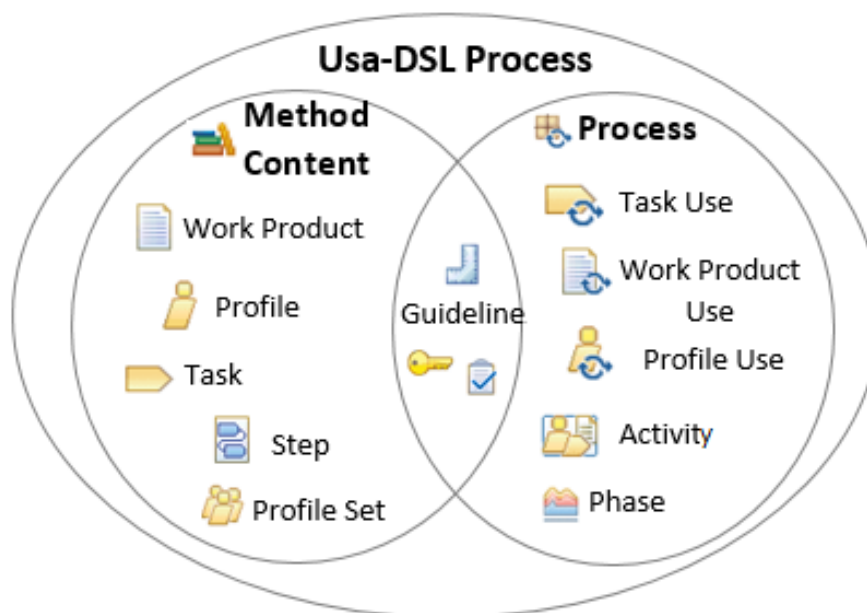


Figure 4.2: SPeM/Usa-DSL Mapping

4.1 Method Content - Usa-DSL Process

The Method Content describes "what is to be produced", the skills needed, and the step-by-step explanation of how the development goals will be achieved. In the Usa-DSL Process, the method content represents "What should be done and who should do it". The Method Content follows the SPeM metamodel mapped to the Usability Framework, which is composed of the elements: Work Product, Profile, Task, and Step. These elements are described next and can be seen in Figure 4.5:

- Work Product - represents data consumed or generated during tasks execution;

- Profile - is an element of Method Content that defines the roles performed within the Process. Profiles are used to define who executes each Task, as well as those responsible for a set of Work Products. There are eight (8) profiles in the Usa-DSL Process, which are grouped into three (3) Profile Sets. The grouping was organized by the type of execution. The main profile sets are and can be seen in Figure 4.3:
 - Process Executors: it encompasses the Usa-DSL users, who plan and conduct the analysis. These profiles are part of the group that designs, develops, and applies the DSL evaluation: DSL Analyst, DSL Developer, and DSL Tester;
 - Usability Evaluation Subjects: it is composed of Domain Analyst, Domain Developer, Domain Tester, and End User. They are those who will contribute to the DSL designers, pointing out the improvements or corrections that must be made to achieve a more pleasant user experience;
 - Heuristic Evaluation Subjects: this Profile is composed only of HCI Expert, the heuristic evaluation expert. They aim to evaluate the DSL and aid the DSL developers to find errors pointed out during the evaluation. They help to improve usability and user experience before presenting the DSL to end users.

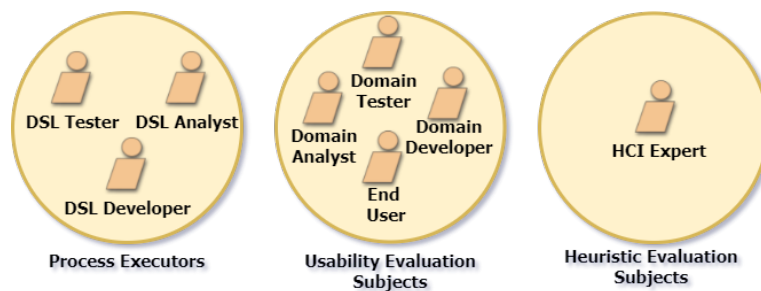


Figure 4.3: Profiles

- Task - describes a work unit assigned to an activity and to a profile, in order to achieve a well-defined goal. It contains complete specification to meet a goal. Work product and guidelines are described in a task to achieve the goal. Tasks are presented in the context of the activity they belong to;
- Step - is a flexible way to define different grouping for content categories. A step is defined by its discipline element, which is derived from the category in the SPEM metamodel. A step is composed of concepts, which are similar concerns and work efforts, that is, it is a collection of Tasks related to certain Activities and executed for a set of profiles called Role Sets. Below we can see the list of Steps that make up the process, the complete description can be found in Section 2.5, Figure Appendix A and an example in Figure 4.4:
 - S01 - Evaluation Profile

- S02 - Ethical and Legal Responsibilities
- S03 - Data Type
- S04 - Empirical Study Method (SE)
- S05 - Evaluation Method (HCI)
- S06 - Metrics
- S07 - Gathering Instruments
- S08 - Evaluation Instructions
- S09 - Evaluation Conduction
- S10 - Data Packaging
- S11 - Evaluation Reporting

Figure 4.4 shows an example of the “Step 1- Evaluation Profile”, phases, tasks, and work products of the Usa-DSL Process.

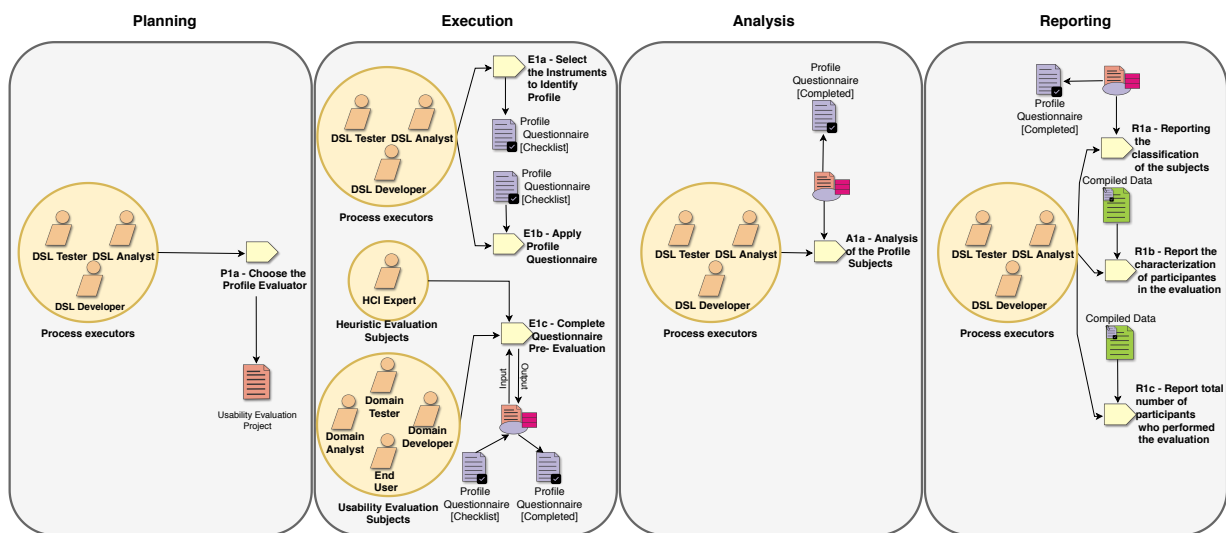


Figure 4.4: The Process - Step 1

4.2 Process - Usa-DSL Process

The Usa-DSL Process provides the elements process life cycle of a usability evaluation derived from the SPEM metamodel. The Process elements in SPEM represent the question "How should be done" in the Usa-DSL. The Process is composed of Phase, Activity, Profile Use, Task Use, and Work Product Use. These elements are described next and can be seen in Figure 4.6.

Usa-DSL Process

Where am I | Tree Sets | Usa-DSL

Discipline Grouping: Steps

A Step is a work grouping based on the similarity of concerns and cooperation of the work effort, that is, it is a collection of Tasks related to certain Activities and composed by a set of profiles called Role Sets.

Categories	<ul style="list-style-type: none"> Usa-DSL
Disciplines	<ul style="list-style-type: none"> S01 - Evaluation Profile S02 - Ethical and Legal Responsibilities S03 - Data Type S04 - Empirical Study Method (SE) S05 - Evaluation Method (HC) S06 - Metrics S07 - Gathering Instruments S08 - Evaluation Instructions S09 - Evaluation Conduction S10 - Data Packaging S11 - Evaluation Reporting

Copyright 2019, Ildevana Poltronieri And others.
All rights reserved

Figure 4.5: Method Content

Usa-DSL Process

Where am I | Tree Sets | Usa-DSL

Usa DSL Life Cycle

Delivery Process: Usa DSL Life Cycle

Description | Work Breakdown Structure | Team Allocation | Work Product Usage

Workflow

The diagram shows a workflow across four phases: Planning, Execution, Analysis, and Reporting. Each phase contains several sub-activities represented by boxes and connected by arrows.

Figure 4.6: Process

- Phase - is an activity type element from the SPEM metamodel. In a phase, the execution of activities is prepared. Phases are oriented by steps that specify which and when activities will be executed. The Usa-DSL Process contains 4 phases: Planning, Execution, Analysis, and Reporting (see Figure 4.4). It is important to emphasize that each process phase is finished when a set of artifacts have been generated by the subjects that have executed the evaluation. Below we can see part of the list of Phases that compose the process, the complete description can be found in Section 2.5:

- Planning Phase

- P1 - Define Evaluators Profile

- P2 - Define Informed Consent Term

P3 - Define Data Type

P4 - Define Empirical Study Method

P5 - Define Evaluation Usability Type

P6 - Define Metrics for Language Evaluation

P7 - Define the Instruments of Data Gathering

P8 - Define the Instruments of Instructions and Training

P9 - Define Execution Place

P10 - Define Data Storage

P11 - Define Study Reporting

- **Activities** - are also activity elements from the SPEM metamodel. Each activity contains a set of tasks that define the basic work unit in the process. An activity represents a general work unit that can be designated to a Profile Use and can have inputs and outputs. It represents a grouping element for other elements, such as Tasks Use, Profiles Use, and Work Products Use. In the Usa-DSL Process, activities group together tasks from a step in a phase from the process life cycle. Hence, each activity contains an identification composed of the initial letter of the phase, a number that identifies a step, and a description of the activity. In Figure 4.4, for example, “E1 - Apply Instruments to Identify Profiles” is an activity from the “(E)xecution Phase” and “Step (1) - Evaluation Profile”;
- **Profile Use** - is the Role Use element from the SPEM metamodel. This element represents the executor or participant of an activity. Profile Uses are defined in the profiles knowledge base. In the Profile Use, the profiles are associated with the tasks they will execute;
- **Task Use** - is a work division element that represents a task that will be used by a specific profile in the context of a specific activity. For example, in Figure 4.4, “P1a - Choose the Profile Evaluator”, which is associated to “P1 - Define Evaluators Profiles” activity will be executed by a DSL designer, for example, Profile DSL Developer, which is part of the profile set from “Process Executors”;
- **Work Product Use** - is an artifact defined in the Work Product and that will be used during the process execution. This artifact can be a product or a document, that can be consumed, generated, or modified during the execution of a task from an activity. For example, the “Profile Questionnaire” is an artifact that must be selected and executed by the profile DSL Developer or any other from the “Process Executors”. In Figure 4.4, this artifact will be consumed in “E1b Apply Profile Questionnaire” and modified in “E1c Complete Questionnaire Pre-Evaluation”.

4.3 Usa-DSL Process Structure

The elements that compose the Usa-DSL Process life cycle are presented in Figure 4.7. Our process life cycle is driven by a set of phases and organized by a set of steps. Each phase is composed of activities and a step is composed of concepts. Both steps and activities are detailed by tasks. A profile performs tasks and is responsible for work products, *i.e.* Guidelines, Checklists, Supporting Materials, Compiled Data, or Templates. These work products are inputs/outputs to/from the tasks.

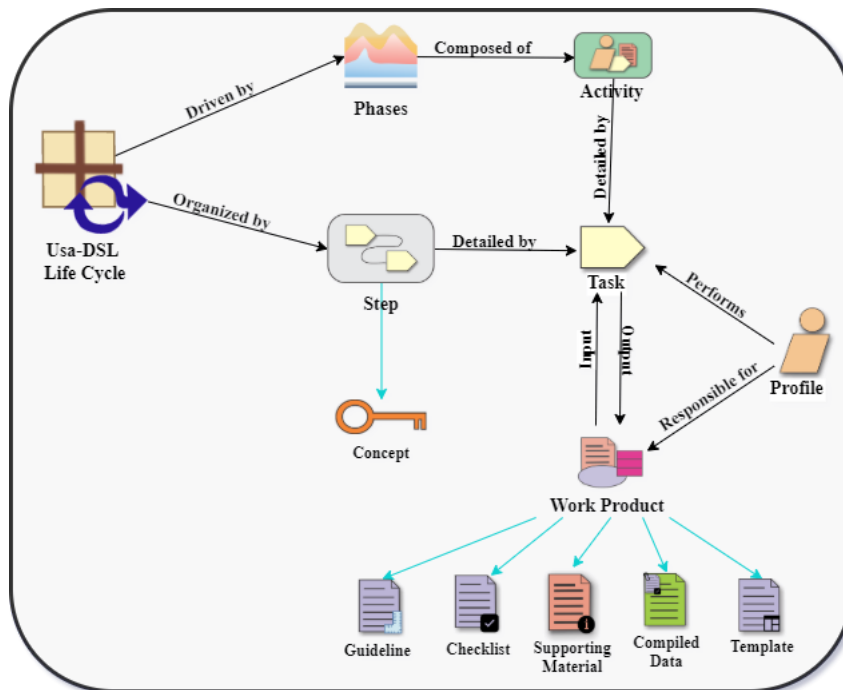


Figure 4.7: Usa-DSL Process Structure

Work products can be guidelines, checklists, or templates. Besides, Usa-DSL Process also provides profiles that execute tasks and are responsible to manipulate work products, for example, modifying templates or support materials, as well as selecting or forwarding checklists to the selected evaluators.

4.4 Usa-DSL Process Life Cycle Modeling BPMN

The Life Cycle represents the question "When should it be done" in the Usa-DSL process. In answer to the question, we represent the execution of the process life cycle using the BPMN notation, supported by the documentation provided by the SPEM metamodel and the Usa-DSL Framework. The choice of the BPMN notation occurred because it allows us greater freedom of representation between Usa-DSL phases and activities without having to

work with the plastered dependencies that SPEM provided us. The BPMN notation provides a graphical representation that makes it possible to present all process flows in a single image/model, as we can see in Figure 4.8.

This modeling was performed in three levels: the first level describes the interactions among the process Phases; the second level describes the interaction among Activities; and, the third level presents the interactions among process Tasks.

- **Level 1 - Usa-DSL Process Phases:** in this diagram, every interaction among the Phases is represented. As can be seen in Figure 2.5, the 4 PEAR phases from the Usa-DSL Framework [62] (see Section 2.5) are represented. From the Planning phase, the usability designer can execute several Activities in this Phase or can go and execute another Activity in the Execution phase. From each Phase, it is possible to execute an Activity in that Phase, go to the next Phase or go back to any previous Phase. Only after executing an Activity on the Reporting phase, the whole process can finish. In this diagram, the elements are used:
 - Pool which refers to the Life Cycle of Phases;
 - Milestones representing each of the Phases;
 - Sub-process that illustrates each of the Phases representing what other activities can be carried out with its expansion;
 - Complex Gateways that indicate that this sub-process can be executed from different flows, such as with iterative and incremental flows;
 - Start event;
 - End event;
 - Sequence Flow Object Connectors connecting each sub-process.
- **Level 2 - Usa-DSL Process Activities:** in this level, there are BPMN diagrams that represent all Activities on each Phase. One simple example could be “E4 - Develop and Conduct Protocol” and “A4 - Analyze the Developed Protocol”. The former belongs to the Execution phase and the latter belongs to the Analysis phase. The diagrams can be seen in Figure 4.8 and in Appendix B;
- **Level 3 - Usa-DSL Process Tasks:** in this level there are BPMN diagrams that represent Tasks interactions for each Activity. Furthermore, these diagrams include the decision elements that determine the Task flows as well as their input and output artifacts, *e.g.* guidelines, checklist, or the messages exchanged among tasks (see in Appendix C). One example would be Activity “E1 - Apply Instruments to Identify Profiles” that is composed of tasks: “E1a - Select the Instruments to Identify Profile”, “E1b - Apply Profile Questionnaire” and “E1c - Complete Questionnaire Pre-Evaluation”.

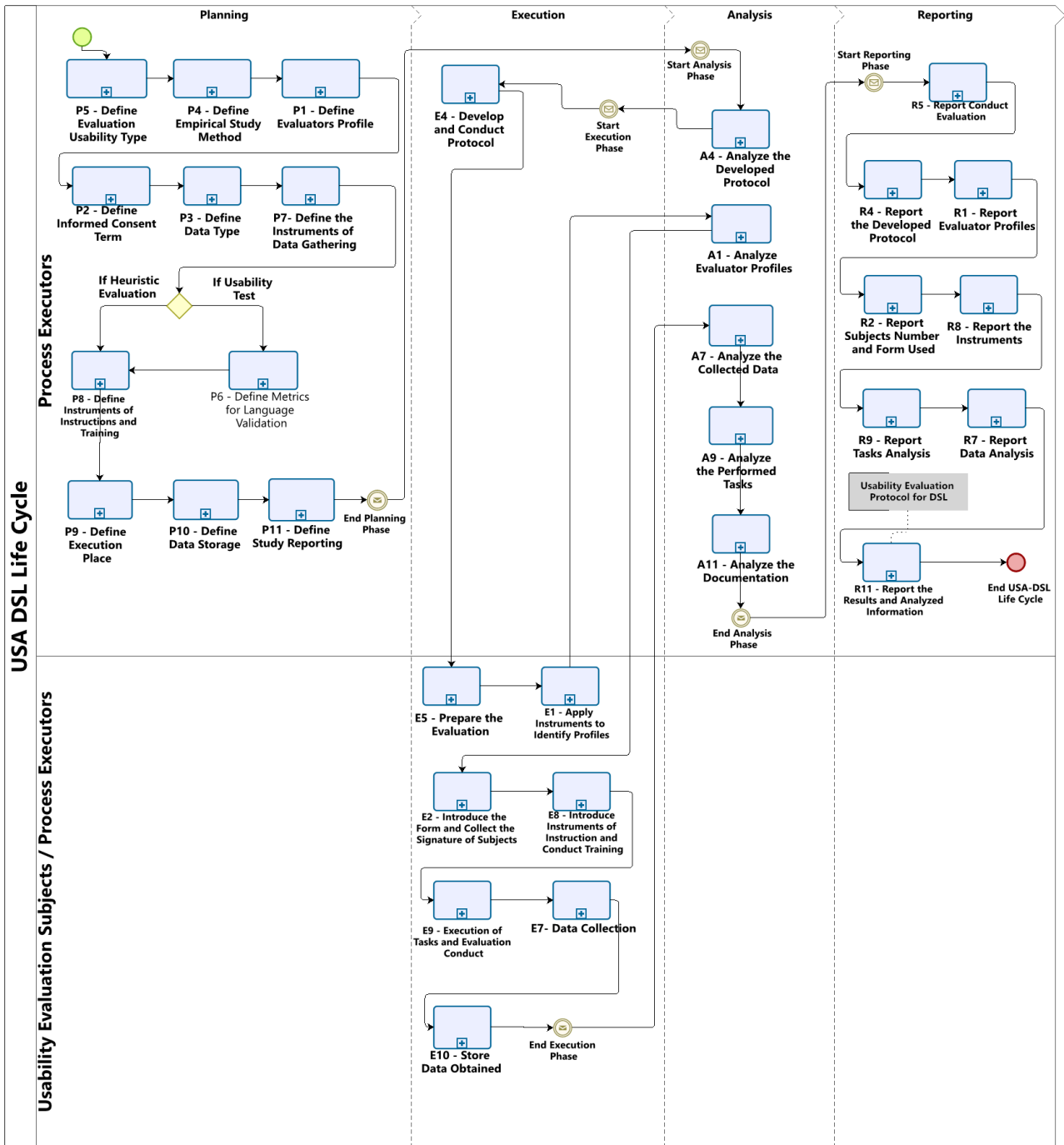


Figure 4.8: Usa-DSL Process Flow BPMN

As we can see, the beginning of the process execution takes place in the phase planning, which occurs sequentially going through and executing each of its activities. In the execution and analysis phase, the activities are not carried out linearly, these two phases execute activities in parallel, that is, they go back and forth between the activities of each phase. In the Report Phase, the process goes back to executing the activities sequentially. The representation presented in Figure 4.8 can help Process Executors, as it shows the flows to be followed and the activities that will be executed in each one. During its execution, process users can click on activities at any time to obtain the guidance and sub-flows between

the tasks they must follow, as can be seen in Figure 4.9 which shows the Activity “P5 - Define the Usability Evaluation” represented by a sub-process “P5a - Select Usability Evaluation” and in Figure 4.10 which presents how the tasks of this sub-process can be performed.

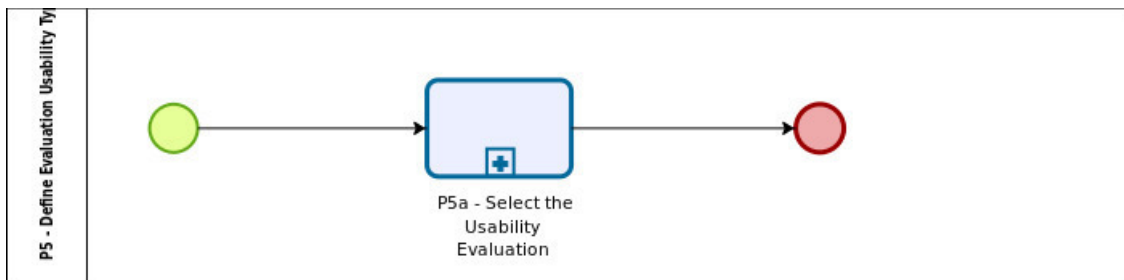


Figure 4.9: Activity P5 - Task P5a Flow

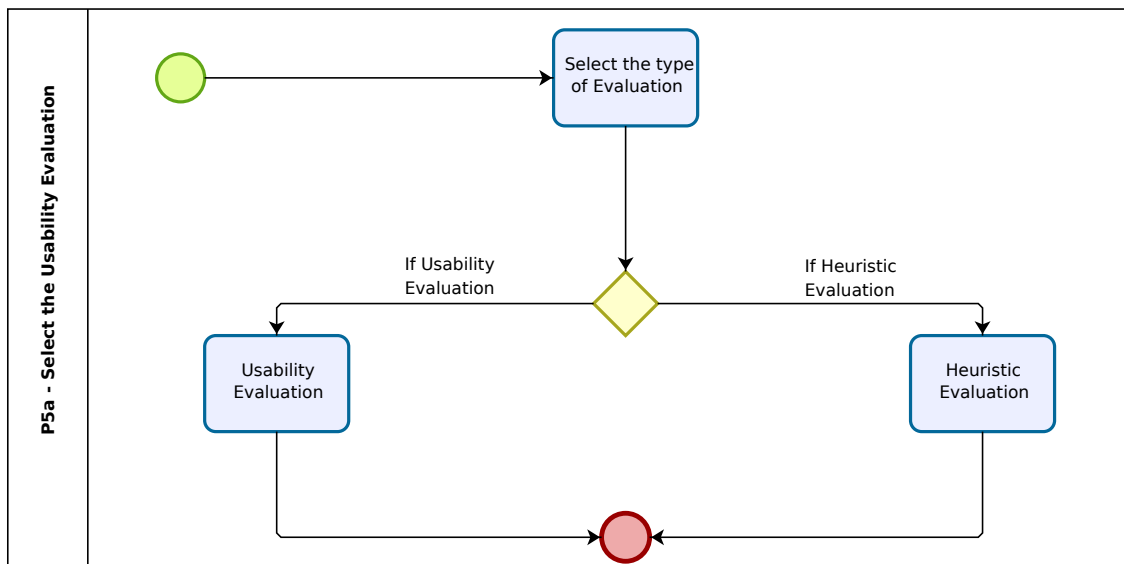


Figure 4.10: Task P5a (expanded)

4.5 Mapping Usa-DSL Process to SPEM

Several elements, or groups of elements, from the SPEM metamodel, are mapped to the Usa-DSL Process. Tables 4.1.a and 4.1.b show this mapping. Notice that some of the terms are slightly modified, thus they maintain the same one used in the Usa-DSL Framework [62].

Regarding SPEM and the Usa-DSL Process Method Content, the mapping presented in Table 4.1.a is:

- Work Product Definition was renamed as Work Product, thus it also maps directly to the Eclipse Process Framework Composer (EPF) [24];

Table 4.1: Mapping Usa-DSL Process to SPEM

Table a. Method Content		Table b. Process	
SPEM	Usa-DSL	SPEM	Usa-DSL
Work Product Definition	Work Product	Work Product Use	Work Product Use
Role Definition	Profile	Role Use	Profile Use
Task Definition	Task	Task Use	Task Use
Category/ Discipline	Step	Activity	Activity
Category/ Role Set	Profile Set	Activity Guidance	Phase Guidelines
Guidances	Guidelines		

- Role Definition was renamed to Profile to keep the Usa-DSL Framework nomenclature;
- Task Definition was renamed to Task to keep the Usa-DSL Framework nomenclature;
- Elements Discipline and Role Set, from the Category element in SPEM, were mapped to Step and Profile Set in the Usa-DSL Process to keep the same nomenclature used in the Usa-DSL Framework.

Regarding Process (Table 4.1.b), the Usa-DSL Process basically maintains the SPEM nomenclature. The difference is Role Use, from SPEM, which is mapped to Profile Use, from the Usa-DSL Framework [62].

It should be noted that the elements that did not follow the SPEM nomenclature are those that had already been provided for in the Usa-DSL Framework. This Framework was defined as conceptual, as it has a high-level view of its elements. For this reason, only some of these are part of the process being developed. In turn, the elements that are being defined from this thesis proposal seek to keep the nomenclature of the SPEM metamodel followed, the structures of the SPEM Metamodel can be seen in Figure 2.7, as well as that of the Usa-DSL Process in Figure 4.2. The changes described above and shown in Table 4.1 can be better located in the diagram in Figure 4.11. For example, the Category element that is represented in UsaDSL by the Steps and Profiles Set sub-elements, as well as the Activity element that is represented by the Phases and Activities sub-elements.

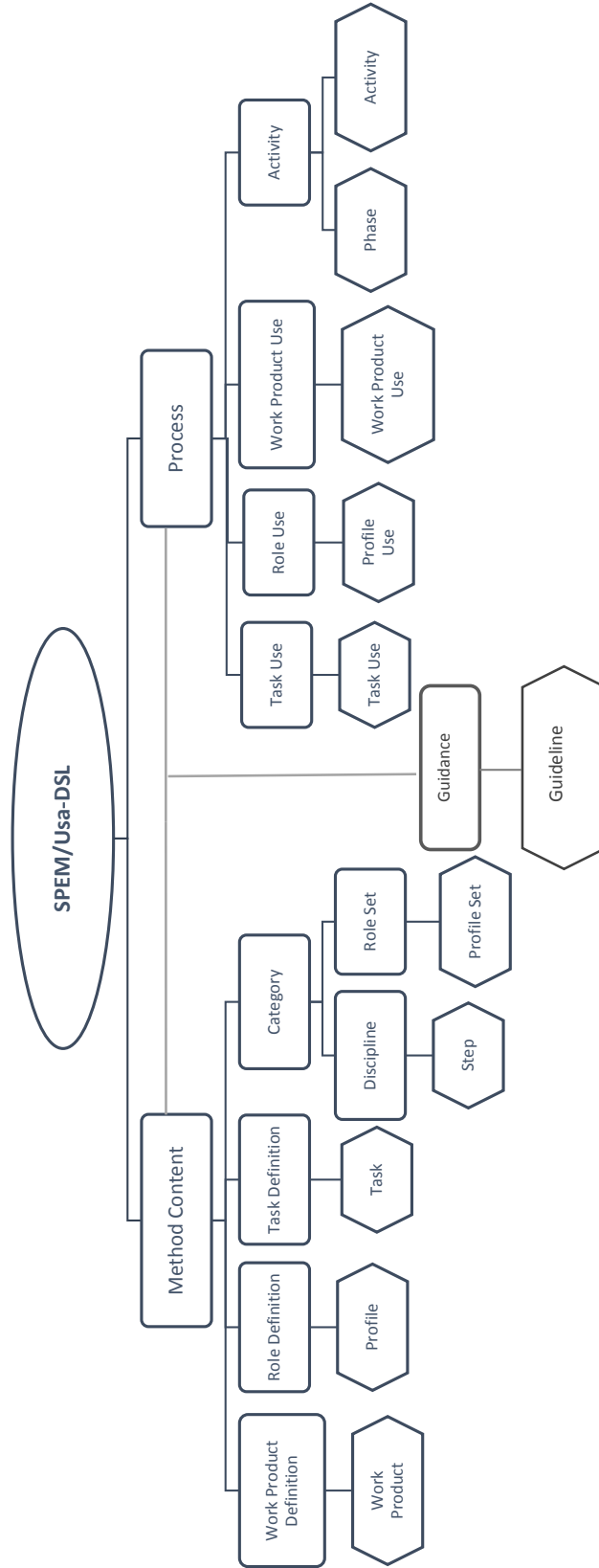


Figure 4.11 : Mapping SPEM/Usa-DSL

4.6 Usa-DSL Process Work Products

A Work Product within Usa-DSL are artifacts consumed, generated or changed during the execution of the usability evaluation. We can conceptualize an artifact as a document that does not appear spontaneously, but that is developed by people/researchers who, based on a need, decide for their construction, whether through a research or work situation. This artifact has a function, form, structure, quality, and meaning within the environment for which it was developed. For this reason, the inclusion of an artifact in a certain environment can change the situation previously experienced, due to its influence on the way it will be used [6].

For this reason, throughout the studies of this thesis, the need to develop and customize artifacts that will help developers in preparing and conducting their evaluation was identified. As previously described, artifacts are part of an environment, in this case, the Usa-DSL Process, and must be used according to the guidelines described in the process. In this section, only artifacts that needed a solid theoretical basis for their elaboration or those that needed adaptation/customization depending on the application domain will be presented.

Among the artifacts that will be presented there are: Heuristic Evaluation Checklist for DSL (Subsection 4.6.1), Usability Questionnaire (Subsection 4.6.2), Profile Questionnaire, and Informed Consent Term (ICT). The other artifacts can be accessed on the Usa-DSL Process¹, as can be seen in Figure 4.12.

The screenshot shows the 'Usa-DSL Process' website. The main navigation menu includes 'Where am I', 'Tree Sets', and 'Usa-DSL'. The 'Usa-DSL' menu is expanded, showing a list of artifacts such as 'Case Study Protocol', 'Compiled Data', 'Controlled Experiment Protocol', 'Data Access Agreement', 'Guide DSL', 'Heuristic Evaluation Checklist for DSL', 'Informed Consent Term', 'Language Presentation', 'Profile Questionnaire', 'Protocol Documentation', 'Protocol Documentation Heuristic Evaluation', 'Protocol Documentation Usability Test', 'Technical Report', 'Scientific Publications', 'Script Interview', 'Survey Protocol', 'Usability Questionnaire', and 'Use Scenario'. The 'Work Products' section is highlighted, and the 'Contents' list is visible.

Figure 4.12: Work Products - Guidance

¹<http://lesse.com.br/usa-dsl/>

4.6.1 Heuristic Evaluation Checklist for DSL

The Heuristic Evaluation (HE) method is a widely used approach for usability inspection, in which a group of evaluators inspects an interface design based on a set composed of ten (10) usability heuristics and a severity score rating from 0 to 4 for each encountered problem [53] [55].

Although heuristic evaluation is frequently used for usability evaluation, these heuristics are used to evaluate user interfaces for many different domains. In some studies, heuristics adjustments are needed to ensure that specific usability issues of certain domains are not overlooked [34]. Several authors use an informal process to develop or adapt usability heuristics and do not follow an established and systematic protocol for heuristic evaluation. In our approach, we **use a set of heuristics to evaluate the usability of applications with specific features, and specific aspects not covered by generic sets of usability heuristics.**

This adaptation was based on questions related to our research domain and the 10 heuristics proposed by Nielsen [55]. One of the main goals of this checklist is to enable teams that are part of different phases of the development process to understand and evaluate their application. In our evaluation, we apply that to DSLs.

This approach not only brings the DSL design team closer to the HE method but also assists the HCI experts who will evaluate the DSL in understanding the problem domain that will be evaluated. For a better understanding of the HE methodology extension, we describe it in the next section.

The methodology to develop our Heuristic Evaluation Checklist follows the methodology proposed by [69] (see Figure 4.13). The first steps were: to understand Heuristic Evaluation and Domain-Specific Language concepts, to adapt existing heuristics for the DSL domain, to produce a set of questions based on a systematic literature review [70], and to create our initial Heuristic Evaluation Checklist. After that, we submitted this preliminary checklist to be evaluated by a set of HCI experts through interviews (Chapter 6). Finally, we got a modified, and final, Heuristic Evaluation Checklist that we applied to an example of use.

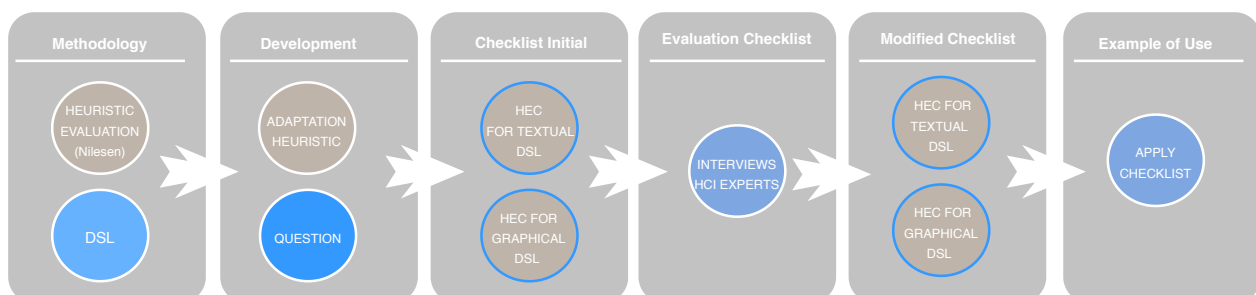


Figure 4.13: Methodology

A Heuristic Evaluation Checklist for Graphical and Textual DSLs

Heuristic Evaluation Checklist (HEC) for graphical and textual DSLs is based on the extension of artifacts from the HE method. This checklist is an artifact designed to guide the heuristic evaluation of DSLs and it must be used in the context of a usability evaluation process for DSLs. This evaluation should be planned by an analyst, developer or DSL tester, and be conducted by heuristic evaluation experts.

The checklist structure consists of five (5) columns (see Table 4.3): the first column contains the identification of the heuristic, the second column is related to the description of the ten (10) Nielsen's heuristics, but adapted to the context of DSLs, the third column refers to the questions that guide the evaluation and are related to each one of the heuristics, the fourth column is the severity degree of the usability problems found and the fifth column is designed for the description of issues found in the DSLs. The Heuristic Evaluation Checklist for graphical and textual DSLs is shown in Appendix D.

This checklist is intended to guide the evaluation of several kinds of DSL. Thus, three distinct versions were created: Heuristic Evaluation Checklist for textual DSLs; Heuristic Evaluation Checklist for graphical DSLs; and, Heuristic Evaluation Checklist for graphical and textual DSLs.

The Heuristic Evaluation Checklist for graphical and textual DSLs cover questions related to both types of DSLs. In this checklist, the first heuristic, "H1: Visibility of system status", for instance, has as description: "The DSL should always keep users informed about what is going on, through appropriate feedback within reasonable time." and it is guided by three distinct questions.

The first question is "Does the graphical DSL provide immediate and adequate feedback on their status for each user action? (For example, after an include or exclude task, the language displays a commit message)?". The second question: "Do the elements available for the user specifically execute only one command? (For example, the 'undo' button only performs undo actions)". The last question (not shown in Table 4.3): "Does the textual DSL provide immediate and adequate feedback on the status of each user action? (For example, after an include or exclude task, the language displays a commit message)?".

The Heuristic Evaluation Checklist for textual DSLs is composed of the same heuristics and descriptions. However, two questions guide the first heuristic on the checklist for textual DSLs: "Does the Textual DSL provide immediate and adequate feedback on their status for each user action? (For example, after an include or exclude task, does the language display a commit message?)" and "Do the elements available for the user specifically execute only one command? (For example, do the keywords on the Textual DSL are used for specific purposes?)".

The same heuristic on the Heuristic Evaluation Checklist for graphical DSLs has also, two guiding questions, which have the same content as the guiding questions on the

Heuristic Evaluation Checklist for textual DSLs but focused on the visual aspects of the DSL. To measure the severity degree of the found usability problems Table 4.2 was used.

Table 4.2: Severity Degree.

Severity	Type	Description
0	Not applicable	I don't agree that this is a usability problem at all
1	Cosmetic problem only	Does not need to be fixed unless extra time is available on the project
2	Minor usability problem	Fixing this should be given low priority
3	Major usability problem	Important to fix, so should be given high priority
4	Usability catastrophe	Imperative to fix this before the product can be released

The use of our checklist is different from previous checklists since each of the heuristics is guided by questions that direct the evaluation to what one seeks to evaluate in a DSL. In the original heuristic evaluation, the evaluators only follow the heuristics and freely point out the errors found when using the system. The Heuristic Evaluation Checklist for DSL will be distributed in three (3) versions: one to evaluate Textual DSL, one to evaluate Graphic DSL and one to evaluate both, making the application domain even more specific, can be seen in Appendix D.

This checklist was directed to a specific need, *i.e.* to evaluate DSLs. Our methodology intends to make the DSL evaluation clearer and more direct, easing the evaluators' task (HCI Experts), even for evaluators that might not be familiar with the DSL domain. Therefore, this study presents, as an example of use, a Heuristic Evaluation Checklist for graphical and textual DSL.

4.6.2 Usability Questionnaire

The Usability Questionnaire for DSL is an artifact developed to be applied in usability testing to evaluate DSL, this instrument will be part of the process instance defined as the Usability Test evaluation. This evaluation will be planned and reviewed by a DSL analyst, developer or tester, as participants in this evaluation, will be invited end-user profiles and DSL experts.

The instrument was guided by Cognitive Dimensions (CD) [31] which are directly related to the metrics described in the Usa-DSL taxonomy [70], [60]. In addition to the CDs and metrics, the instrument defined as Usability Questionnaire for DSL, had the theoretical basis of the A Cognitive Dimensions Questionnaire [17], [16]. For a better understanding of this questionnaire, the concepts that involve each of the cognitive dimensions and their relationship with each of the metrics belonging to the taxonomy will be presented, as well as the structure that composes the questionnaire.

Table 4.3: Snippet for our HEC

Heuristic	Description	Question	Severity (Check each of the problems found)					Description of each error occurrence
			0	1	2	3	4	
H1: Visibility of system status	The DSL should always keep users informed about what is going on, through appropriate feedback within reasonable time.	Does the DSL provide immediate an adequate feedback on its status for each user action? For example, after an include or exclude task the language displays a commit message? Does the DSL provide immediate an adequate feedback on its status for each user action? For example, after an include or exclude task the language displays a commit message?						
		Do the elements available for the user specifically execute only one command? For example, the “undo” button only performs undo actions. Do the elements available for the user specifically execute only one command? For example, the “undo” button only performs undo actions.						

4.6.3 Metrics and Cognitive Dimensions Mapping

The purpose of this mapping is to identify the dimensions that are included among the metrics identified in the taxonomy presented in Figure 3.3, which is based on the terms

mentioned in the selected studies in our 2017 SLR [70] and in the 2021 SLR [60] presented in Chapter 3. Based on the terms of our taxonomy (see Section 3.5) and on the cognitive dimensions (CDs) presented in Section 2.4, mapping was carried out in order to identify the metrics that are related to each of the cognitive dimensions, as can be seen in Figure 4.14.

We can identify that the conciseness metric was the one that most obtained a relationship between the cognitive dimensions, being among these: viscosity, visibility, diffuseness, and hard mental operation. The second most related was productivity which was related to premature commitment, consistency, secondary notation, and progressive evolution CDs. The metric representativeness had four relationships between the CDs, role-expressiveness, abstraction, closeness of mapping and hard mental operation. The metric ease of use was related to three CDs viscosity, visibility and role-expressiveness, as well as the metric re-adaptability that was related to the CDs hidden dependencies, role-expressiveness and provisionality. The metric effectiveness, Usage satisfaction, comprehensibility, perceived complexity and flexibility presented only two relationships.

Effectiveness was associated with the premature commitment and progressive evolution CDs, usage satisfaction with the visibility and abstraction CDs, comprehensibility with the role-expressiveness and abstraction CDs, perceived complexity with the secondary notation and diffuseness CDs, and flexibility with the consistency and provisionally CDs. The other metrics and CDs were related only one to one, such as: metric efficiency was related to CD viscosity, error-rate, and task-error metrics to CD error-properness, metric intuitiveness was related to CD closeness of mapping and metric understand learning with CD hard mental operation.

Usability Questionnaire for DSL

The cognitive dimensions give us indications of being a good tool for evaluating DSLs. They provide terminology on the perception of problems related to notational systems, and are universal enough to allow identifying problems in visual modeling languages. According to the study by Blackwell [17], their perception and the contexts in which they tested it revealed some small problems, such as the generalization of the questionnaire. The authors justify that such problems can be eliminated by customizing the context in which the questionnaire is to be applied. That's what happened in our first study 7, we ended up making adjustments at certain points in the questionnaire, in addition to adding answers on a Likert scale to meet the needs of the language engineer who was using Usa-DSL. The adapted questionnaire can be seen in Appendix F.

Even using the adapted questionnaire, it is noteworthy that we provide a version of the questionnaire as close as possible to its original design, with only a few adaptations. The adaptations were made: in Part I, which removed the information about DSL, language proficiency and use of DSL in the same context, as this information is presented at the

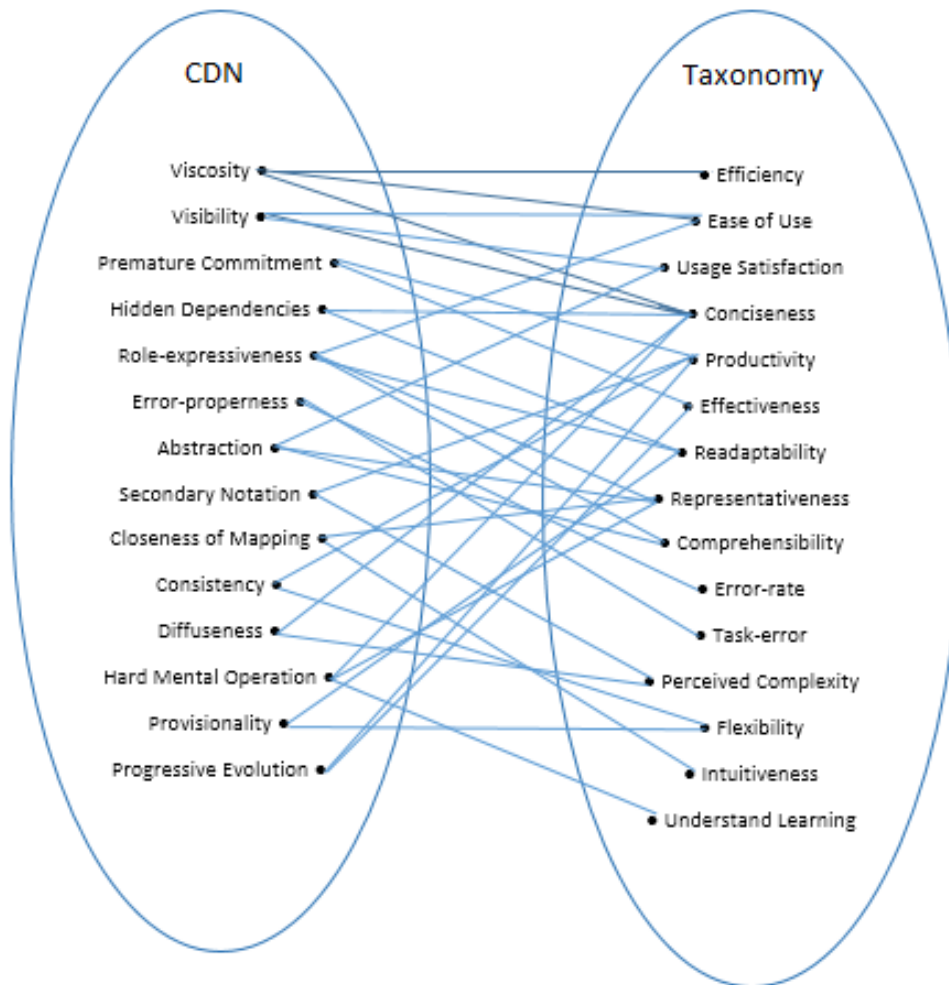


Figure 4.14: CDN and Taxonomy Metrics

beginning of the evaluation; in Part II, which was defined as Part I in which it presents the definitions, we changed the word notation to DSL and the usage example that we use the Cucumber product supported by DSL Gherkin in the context of Behavior-Driven Development (BDD); Part III, was changed to Part II, but keeps the Questions about System, also Part IV, which in our questionnaire refers to Part III, but keeps the Questions about Notation/DSL and Part V was changed to Part IV, keeping the Questions about Sub-devices, as well as all other questions. This questionnaire by was called the Usability Questionnaire and can be seen in Appendix E.

4.6.4 Profile Questionnaire

An important activity in investigation/evaluation is to know and understand who the research participants are. The information collected helped to understand the phenomena

that will be captured at the end of the evaluation, so we need to pay attention to the information we intend to obtain from the participants [6]. For this, a Profile Questionnaire must be used that is able to extract the greatest amount of information about the profile of the subjects that make up the sample, without this being extensive and tiring and causing the participant to give up participation.

The Profile Questionnaire is an instrument that seeks to gather some information such as demographic data, area of expertise, experience in the position held, and even data that provide insights into the knowledge or habits of the participants. The data obtained can help companies, scientists, and academics to identify and profile the people who will be potential research participants. The information obtained is important for determining the sample, as well as delimiting the threats to the study when it has a group that is not homogeneous, or with very different knowledge from what is needed to effectively participate and contribute to the study [86]. Considering the importance of this information about the participants, a Profile Questionnaire is presented in the Usa-DSL Process as a template, the data we seek to collect in our questionnaire are regarding knowledge about DSL, empirical studies and usability evaluation, level of education, area of expertise (Industry/Academic), experience time, among others. In addition to these data, Process Executors can add other questions if they feel the need or are important for the domain of their research. The template we provide can be accessed at the process link and seen in Appendix H of this thesis.

4.6.5 Informed Consent Term

Many areas involve ethical concepts in their performance and regulation. It is noteworthy that all research involving the participation of people is more concerned with ethical aspects. In particular, an area that stands out in this aspect is health, as it directly involves people. Computing also has ethical concerns in its research, including some associations that guide professionals in relation to ethical aspects such as ACM and IEEE. Research carried out in Brazil has Resolution No. 466 of December 12, 2012 [49], which regulates and approves the guidelines and standards for scientific research involving human beings, in any area of knowledge.

In this sense, as our research involves the participation of people, we rely on Resolution No. 466/12 [49] to develop the Usa-DSL Process Informed Consent Term (ICT) document. From this resolution, we highlight two considerations that support our document: 1) “the ethical issues raised by the progress and advancement of science and technology, rooted in all areas of human knowledge; and 2) that all progress and advancement must always respect the dignity, freedom and autonomy of the human being”.

Following Resolution No. 466/12 [49], the ICT is defined by the following terms and definitions: **Research Findings** - the facts or information found by the researcher during the

research must be presented to the participants; **Free and Informed Evaluate** - participants must be informed about the nature of the research, its objectives, methods, expected benefits and potential risks when any, to the extent of their understanding and respected in their singularities; **Research Proponent Institution** - the organization, public or private, to which the responsible researcher or co-participants are linked must be informed; **Research Participant** - the individual who, in an informed and voluntary manner, accepts to be researched or contribute in some way to the study; **Researcher** - a member of the research team, co-responsible for the integrity and well-being of the participants; **Responsible Researcher** - person responsible for coordinating the research and co-responsible for the integrity and well-being of the participants; **Research Protocol** - set of documents that describe the research in its fundamental aspects and information related to the research participant, the qualification of the researchers and all responsible bodies; **Informed Consent Term (ICT)** - document in which the participant's free and informed consent is explained, containing all the necessary information, in clear and objective language, easy to understand, for the most complete clarification of the research in which it is proposed to participate.

In fulfilling the requirements involving research with people, we highlight what was described and explained in the document ICT:

- justification, objectives and procedures that will be used in the research, with details of the methods to be used, informing the possibility of inclusion in a control or experimental group, when applicable;
- explanation of the possible discomforts and risks arising from participation in the research (if any), in addition to the expected benefits of such participation and presentation of measures and precautions to be used to avoid and/or reduce adverse effects and conditions that may cause harm, considering characteristics and research participant context;
- clarification on the form of follow-up and assistance to which research participants will be entitled, including considering benefits and follow-up after the end and/or interruption of the research;
- guarantee of full freedom for the research participant to refuse to participate or withdraw their consent, at any stage of the research, without any penalty;
- ensuring the confidentiality and privacy of research participants is maintained during all phases of the research;
- guarantee that the research participant will receive a copy of the ICT.

In addition to complying with the provisions of this Resolution, it is understood that each area of investigation and each type of research must comply with sectorial requirements

and specific regulations. Therefore, respecting the regulations of the Resolution and the particularities of the area, we seek to help Language Engineers by developing an Informed Consent Term (ICT) guideline that is part of the Usa-DSL Process. This document follows the previously informed guidelines and can be seen in the Appendix G.

4.7 Final Considerations

Although different research present ways to evaluate DSL usability, the Usa-DSL Process, to the best of our knowledge, is the only complete process that helps a DSL designer choose different methods, *i.e.*, experiments, case studies, or surveys, and also usability evaluation methods, *i.e.* Heuristic Evaluation or Usability Testing. The Usa-DSL Process contains evaluation concepts, activities that have to be executed and their respective performers. Furthermore, the DSL designer can also follow a well-defined evaluation life cycle presented by the Usa-DSL Process.

It is very important to avoid possible failures in the deployment of a DSL, hence a DSL designer should execute a well-defined usability evaluation, so, in the end, the DSL user experience is pleasant. In that direction, the Usa-DSL Process provides means to facilitate the planning, execution, analysis and reporting of the usability evaluation of a DSL.

Unlike the related work mentioned, the Usa-DSL Process does not involve only evaluations based on metrics and features to be used but also encompasses the entire building process of an empirical evaluation, using in a clear way the aspects of usability evaluation. This proposal was built with a complete life cycle in mind, in which a formal protocol is developed with all its steps defined from the Empirical Software Engineering area [86] and concerns from Usability Evaluation [66] and Heuristic Evaluation [55].

Part III

Usa-DSL Process Evaluation

5. USA-DSL EVALUATION: SURVEY

To first introduce the Usa-DSL Process to the community, we invited some experienced researchers on DSL to analyze the metrics Perceived Usefulness (PU) and Ease of Use (EoU) using the Technology Acceptance Model (TAM) [23]. We chose this model because the main goal of this study was to obtain the respondent's perception of the process content and presentation. After that, participants went through the Usability Usa-DSL Process website and provided their insights on the process.

5.1 Planning

Before starting the evaluation we provided the participants with the following documents: (i) **Informed Consent Term (ICT) and Profile Questionnaire**: the ICT was presented only to participants that accepted to answers the profile questionnaire. In order to obtain the information regarding the participants' background, the following questions were asked: **Q1** - What is your name?; **Q2** - What is your e-mail?; **Q3** - What is the name of the institution or company you work for?; **Q4** - What is the role that best describes your job?; **Q5** - How long have you been using or designing DSL (in years)?; **Q6** - Where do you live (country)?; (ii) **The Usa-DSL Process Survey Guidelines**: After completing the profile questionnaire, the participant would access the Usa-DSL Process Survey Guidelines, which contains a brief description of the process. Finally, we asked participants to freely explore the Usa-DSL Process and to analyze and give their feedback during a survey; (iii) **Conduction**: We sent an e-mail with the dates that the survey would be available, estimated duration of the survey, information about the number of questions, and informing the participation would be anonymous.

5.2 Pilot Study

Two experts executed the pilot study. Both of them (E.1 and E.2) are lecturers-researchers, and have two and nine years of experience in designing DSLs. They followed a document guiding them on the Usa-DSL Process, then they took a seventeen-question survey ¹.

With regard to PU, they agreed with all questions by answering "Agree" or "Strongly agree" on the provided documentation, process comprehensibility, process elements comprehensibility, and improvement of planning, execution, analysis, and reporting DSL usability

¹Readers can take our survey at <https://doi.org/10.5281/zenodo.3962211>

evaluation. The only question they did not agree upon was the question that asks whether the benefits outweigh the time required, E.1 did not agree and E.2 was neutral.

On the EoU, both experts answered “Agree” or “Strongly agree”, *i.e.* they consider Usa-DSL objective, easy to use for documenting the DSL evaluation process, ease to follow, ease to plan, execute, analyze, and report, ease to follow its documentation, and ease to find the requested information. The only question they did not agree upon was the question on Usa-DSL requiring little effort, in which E.2 agrees and E.1 is neutral.

We also asked three open questions to the experts. One of them regarding their opinion/statement on the Usa-DSL Process for developing and conducting a usability evaluation for DSLs. E.1 agrees that the process might be used for “...any other usability evaluation. It is a nice guide and provides good accessibility to fast checking information”. E.2 mentioned that “It is a very well-defined process... there are clear steps to facilitate its adoption.”.

Another question was related to the adoption of the Usa-DSL Process for DSL usability evaluation. E.1 would adopt it as “it saves a lot of learning time. It is also quite objective and it is quite easy to interact with. The pages had many links to go in any direction. I liked the flowcharts since they gave an overall idea of the section.”. E.2 said he would “definitely adopt it if I had to make an important decision regarding the choice of a long-term use for a DSL”.

The last question was about the recommendation of the Usa-DSL Process to a colleague to support DSL usability evaluation. E.1 mentioned that “I will recommend to my students and collaborators since it can save learning time and can be used as a guide/manual to fast checking the doubts”. E.2 would also recommend it “due to the detailed description and completeness”.

Both DSL designers also suggested improvements to our process. Based on the pilot study, we improved the Usa-DSL Process and invited more experts to evaluate it.

5.3 Result Analysis

Twenty-one (21) participants responded our survey. The participants DSL experience was: ten (10) participants had between 0.5 to 3 years of experience, five (5) participants had 4 years of experience, four (4) participants had 5 years of experience and two (2) participants had been working for 10 years with DSL. Regarding the country of residence, twenty (20) are Brazilian and one (1) is German. Regarding the educational institution or company, the participants indicated UTFPR, UNIPAMPA, DJukic Software GmbH, SETREM, Porthal, among others. Considering their job description: three (3) were developers, one (1) was a software architect, four (4) were professors/researchers and thirteen (13) were master students.

We analyzed **Perceived Usefulness (PU)** based on six questions and 21 respondents in two perspectives: internal cohesion of responses by applying the Cronbach's Alpha [21] analysis, and mode of the responses per respondent and per question.

We applied **Cronbach's alpha to PU** answers for measuring their internal cohesion. Therefore, we came up with $\alpha = 0.8805237$, *i.e.* respondents tend to answer the same way for all PU questions (PU.1 through PU.6). Figure 5.1 summarizes the stacked results regarding each PU question.

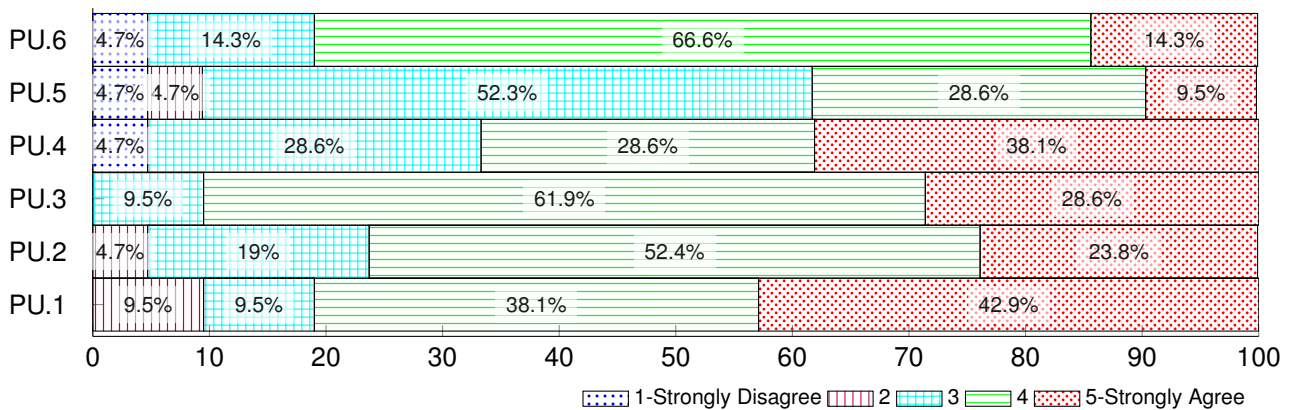


Figure 5.1: Perceived Usefulness Results of Usa-DSL

Analyzing **PU.6**, which represents the overall perceived usefulness of the Usa-DSL Process, we observed that 17 (80.9%) respondents found the Usa-DSL Process useful for performing usability evaluation for DSL, whereas three (14.3%) were neutral. With regard to **PU.1**, 81% of the respondents confirmed that the Usa-DSL Process helps them to understand what should be done for evaluating the usability of a DSL, *i.e.* nine (42.9%) responded "Strongly agree" and eight (38.1%) "Agree". **PU.2** focuses on whether Usa-DSL helps someone to understand how to perform a DSL usability evaluation. In this aspect, 76.2% of respondents "Agree" that the Usa-DSL Process helps, *i.e.* five (23.8%) responded that they "Strongly Agree" and 11 (52.4%) "Agree". With relation to whether it is possible, based on its elements, to understand the Usa-DSL Process (**PU.3**), 90.5% of respondents believe that that is the case, *i.e.* six (28.6%) responded that they strongly agree and 13 (61.9%) agree. 66.7% of respondents confirm that planning, execution, analysis, and report of DSL usability evaluations is improved when using the Usa-DSL Process (**PU.4**), *i.e.* eight (38.1%) responded that they strongly agree and six (28.6%) agree. We also asked respondents whether they found the Usa-DSL Process long, but its benefits outweigh that (**PU.5**). Only 38.1% agreed with that, *i.e.* two (9.5%) strongly agreed and six (28.6%) agreed. This is the least well-evaluated aspect of our process. Nonetheless, 11 (52.3%) respondents were neutral. This might not be conclusive since respondents did not have a chance to use the Usa-DSL Process in an actual DSL usability evaluation.

To analyze the **Ease of Use (EoU)**, first we applied Cronbach's alpha to EoU answers to measure their cohesion. Results showed $\alpha = 0.8370315$, which means that respondents tend to answer the same way for all EoU questions (EoU.1 to EoU.8).

In general, most of the respondents (76.1%) found the Usa-DSL Process ease to use (**EoU.8**). Three participants (14.3%) strongly agreed that Usa-DSL is objective (**EoU.1**), and nine (42.8%) agreed. For Question **EoU.2**, 71.4% agreed that Usa-DSL is easy to be used to fully document elements of a usability evaluation for DSL, *i.e.* five (23.8%) respondents strongly agreed and ten (47.6%) agreed. For Question **EoU.3**, 61.8% of the respondents agreed that the process flow was easy to be used when performing a DSL usability evaluation, *i.e.* four (19%) respondents strongly agreed and nine (42.8%) agreed.

As a matter of facilitating planning, execution, analyzing, and reporting a DSL usability evaluation (Question **EoU.4**), 71.3% agreed that the Usa-DSL Process has this capability, *i.e.* six (28.5%) responded that they strongly agreed and nine (42.8%) of them agreed. Question **EoU.5** asked about how easy is to browse the Usa-DSL process pages and 66.6% of respondents confirmed it was easy, *i.e.* ten (47.6%) participants strongly agreed and four (19.0%) agreed. 80.9% found it easy to find the necessary elements to perform usability evaluation for a DSL using Usa-DSL (Question **EoU.6**), *i.e.* seven (33.3%) participants strongly agreed and ten (47.6%) of them agree. Question **EoU.7** asked whether participants found that interacting with Usa-DSL required little mental effort or not. Eleven (52.3%) participants agreed with that and seven (33.3%) of them were neutral to this question. Again, this might have happened because participants did not use the Usa-DSL Process in an actual DSL usability evaluation. Figure 5.1 shows the stacked results regarding each EoU question.

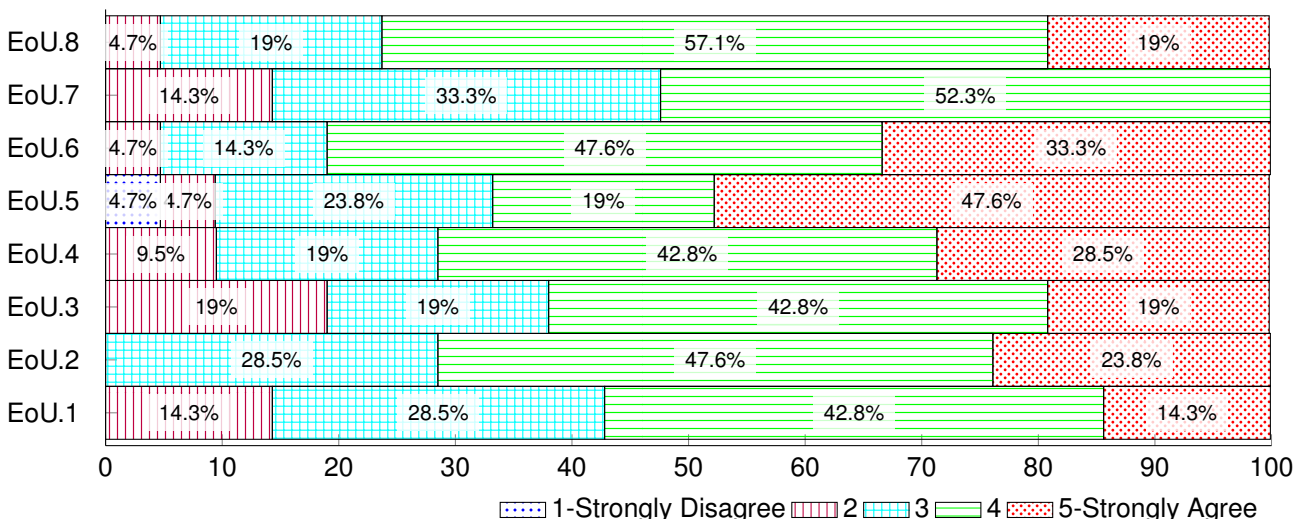


Figure 5.2: Ease of Use Results of Usa-DSL

Open Questions. We adopted Deedose² to help us in organizing respondents quotes and identifying open and axial codes from the three open questions responded by

²<https://www.dedoose.com>

each respondent. Based on what they said, we identified and categorized their words and sentences. As a total, we came up with 11 codes in four categories: **Positive Aspects**, **Aspects to Improve**, **General Usability Evaluation**, and **Prospective Recommendation**. **Guidance** of Usa-DSL was the most mentioned by the respondents, with 15 quotes, followed by **Prospective Recommendation** with 13, **Learning and Usage** with 12, and **Adoption** with 11 quotes. We analyzed the identified categories and respective codes exemplifying them with respondent quotes next.

Positive Aspects. This code represents the aspects that respondents believe would make the Usa-DSL Process a success. **R.7** claimed that Usa-DSL “comes to aid in an important gap in the domain-specific language development process, which is the usability concerns, that feature has been put aside in DSL community”. **R.1** would recommend Usa-DSL to students and collaborators, whereas **R.12** said that “the process seems relevant to be used”.

Learning and Usage: **R.1** affirmed that the process is quite objective and easy to interact with, as well as he/she likes the process flowcharts as they provide the process overall idea. **R.2** corroborates **R.1** as he/she found that the process steps clear, which facilitates its adoption. **R.3** agreed that the process is consistent, corroborating **R.6** who said “process documentation is well detailed”, as well as activities and artifacts, which guide new Usa-DSL users. **R.10** and **R.14** also mentioned the importance of process details and documentation. Presentation and explanation of the process using charts is highlighted by **R.15**, whereas **R.16** emphasized process iterative and incremental characteristics.

Guidance: **R.1** found that the process has nice guidance as it provides good accessibility and fast information checking. **R.2** commented that the process is well defined and complete. **R.3** highlighted that the process seems natural to evaluate DSLs and **R.14** said it is clear who does what. **R.6** and **R.9** focused their quotes on the ease of navigability and intuitiveness throughout its website. **R.8** and **R.16** found that the process could be used in medium and large companies and it adds value to the company solutions.

Adoption: nine respondents said they would definitely adopt Usa-DSL. **R.2** and **R.15** claimed the process clear steps eases its adoption. **R.12** mentioned that “if the usability of a given DSL would be a priority” then the process would help. **R.8** found that in projects with more people, this process would be adopted. **R.7** and **R.10** agree on the adoption of Usa-DSL as it might bring useful feedback from DSL users and could also be a development checklist.

Recommendation: **R.3** would recommend the process if the development of a DSL is the central objective. **R.17** found the process helpful for non-experts on DSL, thus **R.17** would recommend it. **R.19** would also recommend the process.

Aspects to Improve. **R.4** mentioned that the Domain Specific Modeling (DSM) approach and the architecture of the DSM solution are not clearly identified from the Usa-DSL Process and, therefore, should be clearer. Furthermore, **R.1** recommended “to provide links

to use cases examples for each step, phase and activities, as well as planning, execution, analysis and reporting". She/he also suggested that tools should be indicated for each experiment phase.

Conditional Adoption: **R.4**, **R.14**, and **R.16** would consider adopting the Usa-DSL Process, but depends on the context and language. **R.14** said, "Yes, but it depends on the context and complexity of the language domain". **R.4** concluded that he/she might adopt the Usa-DSL Process if the DSM methodology was expressed in a simple way, and **R.16** claimed that "It depends on the scope size, as the application of the process can be very demanding for a small scope". **R.3** would not adopt because he/she did not feel an expert on DSL and **R.18** responded that it is not possible to adopt it if the benefit is not perceived, that is, she/he can only affirm after using the Usa-DSL Process in an actual case.

Evaluation/Validation: all participants mentioned that they could only evaluate after performing/using the process. **R.14** said "I still believe that it is necessary to execute the process in order to be validated" and **R.18** responded that "It is only possible to give a return when the process is used. In the time dedicated to the survey, I believe it is impossible to evaluate the process with only the guidelines".

Massiveness: participants believed that the process is extensive and that for someone who does not have prior knowledge on usability evaluation it still requires some time to be executed. **R.13** said, "I believe that for very small languages it is not necessary, even if it is useful". **R.8** mentioned that "Otherwise I think the process could be very expensive, depending on the scenario, to be carried out in its exact completeness". Meanwhile, **R.3** claimed that "add a considerable layer of work when compared to creating an *ad hoc* questionnaire and testing usability with users".

Navigation: participant **R.6** mentioned that "the documentation helps to ease navigation/analyze", but **R.1** commented that the webpage had many links to go in any direction and **R.17** said "Concerning process website, I think that it is a good idea to integrate the BPMN diagrams with the EPF-Composer framework. However, navigating through the activities diagram was difficult, impacting site usability".

Novice Users: had only one respondent quote, **R.6** said "This version would aid experienced users whom only need to 'remember' or validated few aspects of the process".

Support Tools: two respondents believe it would be good to have supporting tools. **R.15** mentioned that "I believe that with the use of the tool it helps many when checking aspects of usability of DSL" and **R.6** claimed, "Yes, however, it would be good to use a supporting tool as it may reduce the effort of managing the artifacts (reports) generated".

General Usability Evaluation. This category had only one excerpt in which he/she believes that the Usa-DSL Process can be used for any usability evaluation not just for DSL.

Prospective Recommendation. This category has thirteen excerpts, all recommending the Usa-DSL Process to other researchers or colleagues. For example, **R.9** said "I

would recommend it because usability is a crucial factor to ensure the adoption of your DSL according to domain experts with less technical knowledge”; **R.8** said “I would recommend it. By checking the documentation that I accessed it seems to me to be an extremely detailed process, able to provide all the support as a guide in usability evaluations”; **R.12** mentioned that “Likely, the proposed evaluation process is certainly relevant for those interested in usability”; and **R.2** quoted “I recommend it due to the detailed description and completeness”.

5.4 Usa-DSL Process Evolution

Based on the evaluation analysis of the improvements category the next changes to be prioritized in the Usa-DSL Process will be the inclusion of examples of use cases that exemplify each step, phase, and activity. In addition to the availability of links, we aim to develop a tool to support the execution of the process as a whole. Moreover, we also plan to develop usability evaluations involving all steps, phases, activities, and tasks that comprise the process.

Specifically regarding the code Massiveness, we emphasize that the process is executed in flows and that the Process Executors do not need to go through all the entire activities to evaluate its DSL. Instead, the Process Executors choose activities that correspond to the usability evaluation that they want to perform. It is worth noting that the process was designed to be used for users who have no knowledge of usability evaluation. Therefore, it is highly recommended that all process elements store the necessary instructions, step-by-step, on how the user executes them.

In relation to the general evaluation analysis, we understand that most respondents said they would use and indicate the process to their colleagues and other researchers. Nonetheless, a small number of respondents believe that the process should undergo enhancements to be used in all contexts, such as to evaluate simpler DSLs. In this context, we believe that with the development of the process supporting tool, this suggestion will be solved soon.

5.5 Threats to Validity

In this section, we discuss the main threats to the validity of our study and present the strategies we used to mitigate them [86]: (i) **Internal Validity**: To reduce the internal validity of our study, one of the goals for our evaluation was to have a limited number of questions and to use multiple-choice questions as much as possible. This was intended to avoid the participant’s tiredness and to motivate them when answering our questionnaire. Furthermore, we organized our questionnaire into sections to avoid the perception of time

spent responding it. Moreover, we included questions related to the respondents' training and experiences in the instrument. (ii) **External Validity**: In order to have a representative sample in the evaluation, we provided an open questionnaire that could be answered by DSL researchers and practitioners in a convenient way. We had 21 DSL designers respondents that had, on average, 3.5 years of experience on DSL development. We believe that the obtained sample is not as representative as we expected, but they gave some good insights into our process. (iii) **Construct Validity**: To avoid instrumentation problems, we adopted PU and EoU from TAM for organizing and preparing the questionnaire. Besides, we conducted a pilot test to previously validate our questionnaire, provide further adjustments on the clarity of the questions, and anticipate possible issues. (iv) **Conclusion Validity**: Even though obtained data are important to establish that Usa-DSL Process is viable, these results cannot be generalized. Nonetheless, the results provided empirical evidence that our process is designed correctly.

5.6 Final Considerations

This chapter provides an evaluation of the Usa-DSL Process through the Technology Acceptance Model (TAM) [23]. This model helps verify the perception of ease of use and the perceived usefulness of our process by the DSL designers. We consider it important to avoid possible failures in the DSLs developed. For this reason, DSL designers should execute a well-defined usability evaluation, in the end, the DSL user experience is pleasant.

Therefore, to analyze the feasibility of the Usa-DSL we performed a qualitative study based on a survey with 21 experienced participants who freely used and explored our process to provide feedback on the usefulness and ease of use dimensions. Results demonstrated that our process is feasible to evaluate DSL usability, but we need to improve in order to ease its adoption.

In that direction, the Usa-DSL process provides means to facilitate the planning, execution, analysis, and reporting of the usability evaluation of a DSL.

6. HEURISTIC EVALUATION CHECKLIST: INTERVIEW

The Heuristic Evaluation (HE) method is a widely used approach for usability inspection. This method is easy to perform and it allows the discovering of various usability issues. The HE method is freely applied by the evaluators. These evaluators go through the application interface pointing out the errors, and consequently classifying them to the degree of severity.

Although Heuristic Evaluation is frequently used for usability evaluation, heuristics are used to evaluate user interfaces for many different domains. For this reason, many researchers adapt heuristics to their application domain. Several authors use an informal process to develop or adapt usability heuristics and do not follow an established and systematic protocol for heuristic evaluation.

Our approach presents a different strategy to apply HE. The proposal provides a checklist that is different from existing solutions. In our proposal, each heuristic is guided by questions that direct the evaluator to effectively evaluate the DSL.

Regarding the evaluation of the created HEC, we performed 7 (seven) interviews with researchers and professionals on HCI. These interviews were analyzed using the Inductive Thematic Analysis method [20], which resulted in a group of common themes of opinions, suggestions, and ideas discussed in this study. These results led us to make improvements in the HEC.

6.1 Interview Context

To evaluate the Heuristic Evaluation Checklist for graphical and textual DSL we used the qualitative analysis approach that was performed through online interviews. The Interview Method [20] was chosen because the main goal of this study was to obtain the respondents' perception of the checklist content and presentation. The interviewees were invited from participants' references, thus ensuring that they were experts on HCI or at least had seen usability evaluation before.

Before the interviews, the documents necessary to the understanding of the analyzed domain, *i.e.* DSLs, were sent to the interviewees, and also the tasks that would be performed were clarified. Thereafter, the participants needed to perform the analysis of the checklist and to provide their contribution during the interview.

The list of documents that were provided to the experts before the interviews were *Informed Consent Term (ICT) and Profile Questionnaire, Survey Guideline, and Information on the Interview.*

6.2 Pilot Test

The pilot test was a small trial to assure that the study was viable. This test checked whether the procedure and questionnaire questions were set properly, and to identify if the process and documents had any potential problem. Furthermore, during the pilot test, small adjustments were made to the main study documents and procedures. To validate the protocol and interview documents, we performed a pilot test with an HCI expert.

During the pilot test, the pilot subject had access to the documents in the same way that actual participants would have, in order to obtain a faithful perception of what would be carried out during the interviews. We verified the duration of the test, the understanding of the messages sent by e-mail, the level of knowledge on the topic to be evaluated, and, finally, the analysis of the checklist.

The pilot subject suggested some improvements on the checklist, for example, to add some examples that include a text box at the end of the instrument, so that the HCI expert could mention problems that would not fit the issues present in the questionnaire. In general, the pilot subject believed that the instructions on the invitation e-mail were clear and met the purpose of the study, as well as the tasks and DSL examples used to accomplish the tasks.

6.3 Profile

For the interviews, seven (7) participants were recruited by e-mail. This sample was selected looking for experienced professionals in HCI (researchers that published relevant papers on HCI conferences). Furthermore, participants were also invited when recommended by other participants who were considered experts. After acceptance, the documents for the interview were sent to them. The Profile Questionnaire was used to identify the experience of the participant and other relevant information. In order to obtain the information regarding the participants' experience, the following questions were asked: **Q1** - What is your name?; **Q2** - What is your work position?; **Q3** - Which usability evaluation method(s) have you already participated in (Heuristic Evaluation, Usability Testing, or None)?; **Q4** - Which usability evaluation method(s) have you already conducted (Heuristic Evaluation, Usability Testing or None)?; **Q5** - What is your level of expertise related to HCI (Very Poor, Poor, Neutral, Strong or Very Strong)?; **Q6** - What is your level of expertise related to Usability (Very Poor, Poor, Neutral, Strong or Very Strong)?; and, **Q7** - What is your level of expertise related to Heuristic Evaluation (Very Poor, Poor, Neutral, Strong or Very Strong)?

In this research the answers of the profile questionnaire, as well as the other answers of the study were identified by the label attributed to each participant, *i.e.* from P1 to P7 (see Table 6.1).

Table 6.1: Subjects Profile

Q1	Q2	Q3			Q4			Q5	Q6	Q7
		HE	UT	None	HE	UT	None			
P1	Professor HCI field	X	X		X	X		V	V	V
P2	Quality Assurance Engineer	X	X		X			S	N	N
P3	Professor/Developer			X			X	N	N	N
P4	Professor HCI field		X			X		V	S	S
P5	Professor HCI field	X	X		X	X		S	N	N
P6	Professor HCI field	X	X		X	X		V	S	S
P7	PhD.Candidate in Computer Science	X	X		X	X		S	S	V

HE - Heuristic Evaluation, **UT** - Usability Testing **V** - Very Strong, **S** - Strong, **N** - Neutral

In this study, as mentioned before, the participants were experts on HCI and most of them had already performed a Heuristic Evaluation. One participant had no experience on Heuristic Evaluation but had a strong level of expertise on HCI, and his consideration of the checklist was relevant on the view of a DSL designer. The other participant who had no knowledge on Heuristic Evaluation had experience on Usability Testing and a very deep knowledge on HCI. The reported experience could be perceived from the responses captured during the interviews.

6.4 Interviews

The interviews started after the pilot test and after each participant had submitted the ICT. The interviews were conducted over a period of six (6) months between December 2018 and June 2019. The execution of the interviews were predominantly online (5 online and 2 in-person). All interviews were audio recorded in order to perform further analysis. Each interview lasted an average of 60 minutes. The interviews were semi-structured, providing a certain flexibility to adjust questioning based on participant responses. Each interview covered five central topics: (1) Definition of heuristics; (2) Checklist's structure and organization; (3) Checklist's content; (4) Amount of information displayed; (5) Checklist's template.

Each topic is guided by questions that direct the interview purpose:

- **Topic 1** - the first topic was assessed by two questions: **T1Q1** - Are the heuristic definitions appropriate for assessing a DSL? **T1Q2** - Is it possible to understand the objective of each heuristic after reading its respective definition?
- **Topic 2** - the second topic was assessed by three questions: **T2Q1** - Does the order in which heuristics are organized adequate? **T2Q2** - Do you think heuristics should be

grouped, for example, in relation to graphical aspects or documentation? **T2Q3** - If so, which heuristics should appear at the beginning or end of the checklist?

- **Topic 3** - the third topic was assessed by five questions: **T3Q1** - Do the questions that correspond to each of the heuristics reflect their purpose? **T3Q2** - In your opinion, are the above questions linked to heuristics clearly and unambiguously? **T3Q3** - Is there any question you do not understand? **T3Q4** - Would you add any questions? Which ones? **T3Q5** - Would you remove any questions? Which ones?
- **Topic 4** - the fourth topic was assessed by one question: **T4Q1** - The checklist is guided by 10 heuristics that are composed of around 32 questions in the most extensive Checklist. Regarding the extension of the checklist, what is your opinion?
- **Topic 5** - the last topic was assessed by one question: **T5Q1** - What do you think about the way the checklist is presented? (Heuristics, Definitions, Questions, Degree of Severity and description of found errors).

After the interviews were completed, the opinions' transcription were performed and organized according to the Inductive Thematic Analysis method [18]. These analyzes are presented in the next section.

6.4.1 Qualitative Analysis

The interview analyses was performed using the Inductive Thematic Analysis method [18], which categorizes the main themes gathered from the experts' responses. This approach is common in HCI qualitative research [20] [45].

For the Inductive Thematic Analysis execution, the audios from the interviews were transcribed. Then, the content was coded by similarity, forming group themes. As the last step, two (2) researchers reviewed the created themes, making some adjustments to best represent the obtained information. The analysis is presented next.

Each theme has a summary description and quotes that support the theme's objective.

Checklist's Description: This theme presents the opinions related to the descriptions used in the heuristic checklist. As the proposed checklist is an instrument customized to contemplate all Nielsen's Heuristics focusing on the DSL domain, one of our main concerns was if the description of the heuristics was appropriated. All participants affirmed that the descriptions created would be useful for the checklist's execution. Participants P1 and P7, who have experience in HE and UT, highlighted the following:

"I think the definitions are appropriated. They are embracing and generic enough, following the pattern created by Nielsen but with a focus on DSLs." [P7]

“It is going to be natural for the evaluators to execute the evaluation as they can evaluate in a broad perspective and make annotations.” [P1]

Detailing: Five participants (P1, P2, P3, P4, and P5) reported the need for more details related to the checklist presentation and also to its content. The main issues reported are related to the severity a rating usage and the lack of questions in specific heuristics.

“It would be helpful to have a small text explaining the severity rating.” [P3]

“The evaluator needs to describe the error in a clear way in order to assimilate the severity rating.” [P4]

“There should be more questions about consistency. I notice that there are some questions related to patterns, but there is a lack of questions about consistency.” [P2]

The above quotes highlight some improvements to be made to the checklist. While developing the customized checklist, we tried to avoid extensive texts in order to not cause fatigue when the evaluation process occurs. However, the feedback received from HCI experts emphasized the need for clarification on certain aspects of the checklist, such as the purpose of the severity classification and the extent of the questions.

Incomprehension: Four (4) participants (P1, P2, P4, and P6) made suggestions for improvements to the checklist or its structure. The suggestions were related to Heuristic 7 (Flexibility and Efficiency of Use) and on the execution of the checklist according to the severity of the classification provided rating.

“I see the severity rating here, but imagine that I did not find any error... What should I mark?” [P6]

The feedback analysis provided by the HCI experts led us to ask how the evaluators would perform the DSL evaluation using this checklist.

Participant P1 reported that the classification of the provided severity scale was not intuitive to use, arguing that it was difficult to understand its use. We mitigated this issue by applying the suggestions captured in the previous theme (Detailing) and adding a small text to guide the severity scale classification.

Another interesting feedback collected was related to Q3 and Q4 of Heuristics 7, as some evaluators disagreed on their answers. The statements of the HCI experts underline the need for minor modifications to make their purpose clearer.

Evaluator’s Profile: Although Participant P3 was not an expert in usability evaluation, he had considerable experience in software development. Hence, P3 stated that the evaluation using the proposed checklist may be performed by any professional with a background in system evaluation, even if this professional is not an expert in heuristic evaluation.

“I think it could be used by evaluators who are not experts in heuristic evaluations but have already evaluated systems using other methods.” [P3]

The statement above highlights the adaptability of this instrument for a wide-range of professionals related to usability.

Content Changes: Two participants (P1 and P4) suggested changes to the content of the heuristic checklist. In general, they stated that some questions needed to be reformulated and/or classified in another heuristic.

“Questions 3, 4 and 5 of Heuristic 3 must be classified in Heuristic 5” [P1]

“Well, I think that the question presented in Heuristic 10 needs to be reviewed.” [P4]

“When you ask about help and documentation... I don’t think that documentation is important for this kind of evaluation, when I read the documentation I think about broad documentation about the system” [P4]

Template Changes: All participants reported the need of changing the order of the visual components that compose the checklist form. The main suggestions were related to the position of the elements and the lack of space for adding relevant information.

“I think that you could use colors to enhance the reading of the heuristics” [P5]

“Maybe the severity description could be at the beginning of the checklist, not at the end as it is. In this way, the access for this information will be easier” [P7]

“It would be interesting to have a blank space for the evaluator to add some relevant issues” [P3]

In order to mitigate these issues, we reviewed the checklist’s template and followed the suggestions of some participants, such as changing the order of problem description and severity rating, so that the checklist would be more intuitive to use, and adding a blank box for the evaluator to describe errors that were not contemplated by the used heuristics.

Instrument’s Amplitude: Participant P7 pointed out one weakness of the checklist by reporting that if evaluators follow narrowly the questions presented in the checklist, maybe some errors presented in the DSL would not be found. We are aware of this weakness and we mitigated it by emphasizing that the role of the questions are to guide the evaluators on common usability problems and the evaluators must report other perceived problems in the extra blank box.

“Perhaps, I don’t know if the evaluators would find all the errors if they followed only your questions.” [P7]

6.4.2 Discussion

This study presents the development of an HEC in which the checklist is guided by questions that conduct the evaluation for a specific domain.

The analyses performed on the interviews showed that this HEC can assist evaluators to conduct DSL evaluations. Furthermore, it was also noticed that a checklist guided by questions directed to the context of use provides a better understanding about the evaluation.

Some changes proposed by the HCI experts were applied to the final HEC (presented in Section 4.6.1). These changes were related to the content of the checklists, as well as, their structure and template. Such changes were pertinent so that it was possible to carry out the first study through an example of use (see Section 6.5), in which a textual DSL was analyzed. The purpose of performing the evaluation of this example of use was to get insights into the proposed Heuristic Evaluation Checklists for DSLs.

6.5 Example of Use

In order to verify the applicability of the proposed HEC, we asked five subjects to experiment it on an example of use. The example of use is a well-known DSL in the academic environment, *i.e.*, LaTeX. This language was chosen since all the participants had previous experience in using with it. The following artifacts were chosen for this example of use: informed consent term, participants profile questionnaire, DSL guide, list of tasks to be performed, and a copy of the Heuristic Evaluation Checklist for textual DSLs (Appendix D).

It is worth mentioning that the main objective of this study was to obtain a first view of the feasibility of the Heuristic Evaluation Checklist for textual DSL. Thus, we only sent the documentation via e-mail and collected the participants' perceptions regarding the heuristic evaluation of LaTeX through our checklist proposal. Hence, from the responses from the participants, we performed a qualitative analysis regarding each of the found problems and their severity.

6.5.1 Analysis

Before discussing the results for each of the ten heuristics in our HEC, we describe the participants profile: one is an undergraduate student and four are master students in Software Engineering; the average time of experience using or designing DSL is 2.4 years; all participants have experience in performing or participating in usability evaluation; three participants have just one year of experience in usability evaluation; two of them have already participated in usability evaluation using usability testing; two of them have already participated in usability evaluation using heuristic evaluation; just one of the participants has already conducted a usability evaluation using heuristic evaluation.

The analysis of the results points for each heuristic as follows:

- H1** Q2: Participants E2, E3, and E5 agreed that there is no undo button and mentioned that the action is only possible using the Ctrl + Z keys. Regarding the degree of severity, two of the participants believed that fixing this should be given low priority.
- H2** Q3: Participant E5 considered that LaTeX has abbreviated keywords and that this makes it difficult for other users to adopt this language. He also reported that it is important to fix this problem, *i.e.* it should be given high priority.
- H3** Q6: Participant E3 reported that some errors are shown in real time and others only after compilation. The participant also indicated that LaTeX does not provide information about the commit, and this may be part of the tool that instantiates the language. Regarding the degree of severity for this problem, the participant considered it a cosmetic problem.
- Q7: Regarding H3, four participants (E1, E2, E3, and E5) reported that there are synchronization problems, the environment warns that changes are made in a certain period of time and that they may not have been saved. Furthermore, they also stated that the changes are saved automatically, but if there is internet connection problems, the re-connection message may not be accurate and changes in the document might not have been saved. E1 mentions: "The problems are only showed after the .tex compilation". This question had a disagreement of the degree of severity among participants, *i.e.* 1, 2 and 4. Hence, there was no consensus among them related to this question.
- H4** No problem found.
- H5** Q11: Participants E1, E4, and E5 mentioned that the environment does not have confirmation boxes or buttons for actions. E1 assigned severity degree 2, while E4 and E5 assigned severity degree 1.
- H6** No problem found.
- H7** Q16: The participants mentioned the following for this question: E2 mentions that if someone considers the generated pdf as an output, in this case, the changes do occur; E4 states that there is no graphic DSL, only the preview of the text written in the generated pdf; and, E5 considers that only when the changes to the textual DSL are compiled, they are observed in the graphical DSL. The severity degree assigned is zero, so they do not consider any usability problem here.
- Q19: Participants E2 and E3 mentioned that to have a color change, LaTeX commands must be entered. These participants considered that this is not a usability problem and attributed zero to the degree of severity.
- H8** Q20: Participants E2, E3, and E4 stated that the error messages are not intuitive or easy to understand, on the contrary, they are confusing and hinder rather than help. Such messages are difficult to quickly identify the problem, so it is often necessary to

have technical knowledge to deal with errors. Therefore, E2 assigned 3 to the degree of severity, while E3 and E4 assigned 2 to the degree of severity.

H9 Q21: Participants E1, E2, and E5 mentioned that there is no tutorial for LaTeX, however, the templates have it. Moreover, related to the degree of severity attributed by those who say there is no documentation ranges from cosmetics to usability catastrophe.

6.6 Final Considerations

This chapter presents our approach to apply HE. The proposal provides a checklist that is different from existing solutions. In our proposal, each heuristic is guided by questions that direct the evaluator to effectively evaluate the DSL.

Regarding the evaluation of the created HEC, we performed 7 (seven) interviews with researchers and professionals on HCI. These interviews were analyzed using the Inductive Thematic Analysis method, which resulted in a group of common themes of opinions, suggestions, and ideas discussed in this study.

As a result, we can see that our Heuristic Evaluation Checklist (HCE) helped the participants to respond more adequately to the questions regarding the evaluation of the DSL. The ones that had previous experience on the heuristic evaluation performed better than the ones that did not have previous experience.

Hence, it seems that, despite our HEC helping the participants to evaluate a DSL, it needs some further instructions on how to fill the questionnaires. Maybe this can help not only experienced participants to better understand and answer the questions. In general, the participants understood what needs to be performed and were able to adequately answer the questions presented in our HEC.

Part IV

Usa-DSL Process Example of Use

7. USA-DSL PROCESS: EXAMPLE OF USE

In this chapter, two evaluations conducted after finishing the Usa-DSL Process will be presented. These evaluations aim to evaluate the Teasy Framework [42], that is, its DSL and the automated sequence generation tool and functional test scripts. It is important to notice that the DSL designer used the Usa-DSL Process to conduct the two evaluations. To identify each of the usability conducted evaluations, we called the first Study 1 and the second Study 2. Study 1 used the usability evaluation method “Usability Test”, which was performed using the empirical method “Quasi-Experiment”. Study 2 used the usability evaluation method “Heuristic Evaluation”, which was performed using the empirical method “Survey”. Usability Evaluation used different workflows in the Usa-DSL Process life cycle, as evaluations have particularities such as participants, activities, tasks, and work products. For a better understanding of the workflow followed in each of the evaluations, its diagrams will be presented: Study 1 can be seen in Figure 7.1 and Study 2 in Figure 7.2. Furthermore, this chapter also presents the evaluations as a whole and the results obtained through the evaluation used instruments. Finally, the opinion of the DSL developers is presented. Their opinions were obtained through a questionnaire to understand their perception regarding the use of the Usa-DSL Process.

7.1 Study 1: Usability Test

In this section, the execution of Study 1 performed to evaluate Teasy DSL will be presented. First, we will present the followed workflow, the performed activities and the used artifacts, in order to understand how the Process Executor conducted the evaluation using the Usa-DSL Process.

Life Cycle: this evaluation followed the life cycle shown in Figure 7.1, that is, guided by the planning, execution, analysis, and reporting phases. This life cycle presents the activities in the order in which they were performed. We emphasize that the evaluation is also guided by Steps that define and detail the tasks that must be performed, as well as the profile that must be performed (this can be seen at <http://lesse.com.br/usa-dsl/> of the process as the Process Executor accesses activities). Each of the tasks has an artifact/work product that is consumed, generated, or changed when conducting the evaluation.

Execution Flow: the used flow to conduct the evaluation was Usability Test. As we can see the evaluation starts with the **Planning Phase**, the first activity to be executed is P5, in which the usability evaluation method “Usability Evaluation” was defined. Then, in Activity P4, the empirical study was defined, among the options, a “Quasi-experiment” was chosen. This method was chosen because it is the most suitable to meet the objective of the

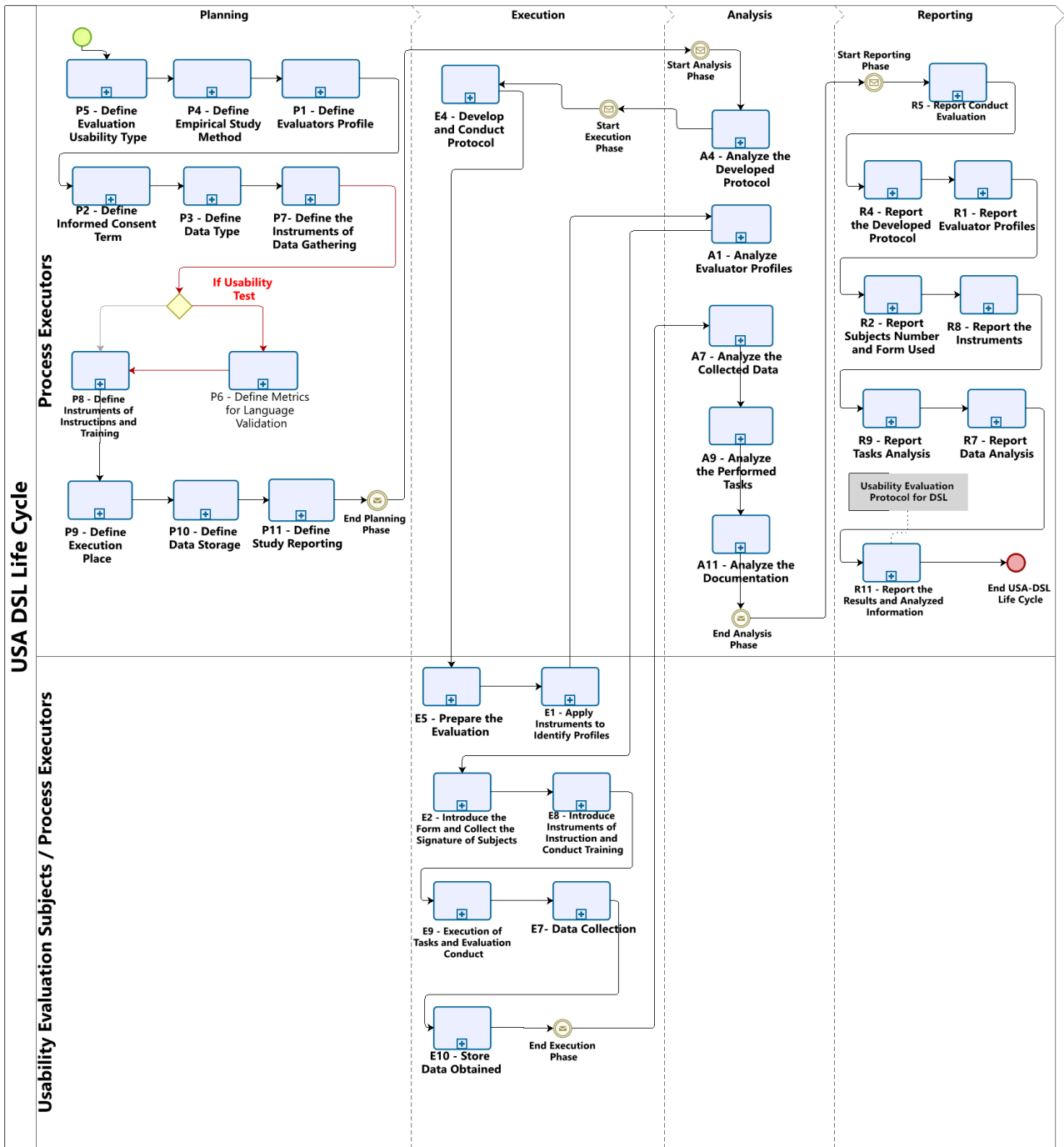


Figure 7.1: Usability Test Workflow

study, which was to compare the Teasy Framework [42] with Robot Framework [15], both for functional testing. Another important point in choosing the quasi-experiment was the current situation of Covid-19, as we would not be able to place participants in rooms (controlled environment) to obtain control over the variables as in a controlled experiment, for this reason, we needed a more flexible method in terms of control or that would not affect the outcome of the study as much. For the execution of this evaluation, the defined profiles in activity P1 were the Experts in the functional test domain. In Activity P2, the Process Executor filled in the data of the study in the Informed Consent Term template. In P3 he defined the Data Type, which

are Qualitative and Quantitative, although our Usability Questionnaire does not have the option of quantitative data, this moment was of great reflection and study both on our part and on the part of the DSL developer. Justifying this moment of study and exchanges, we tried to understand the position of the developer who had no experience and no time to analyze purely qualitative data and asked us for permission to change the Usability Questionnaire so that there was the possibility of extracting quantitative data. At this point, we focus on adjusting the questionnaire to meet the user's needs (in this case the Process Executor). In Activity P7, the instruments to perform the Teasy evaluation were selected and changed or developed. In this activity, the Process Executor recorded the instructional videos for using the DSL, created the documents with the tasks that would be performed, and changed some questions in the Profile Questionnaire to raise the profile of the participants. In Activity P6, which defined the metrics and the Usability Test instrument, the metrics effort, time, and completed tasks were selected, in addition to using the Usability Questionnaire adapted from the Cognitive Dimensions Questionnaire, which for this evaluation we decided to test it in a partial and adapted way (such as explained above). In this case, Activities P9 and P10 were defined for convenience in the online format. As this process is iterative and incremental, activities are executed among phases and can go from Planning to Analysis, as we can see in Figure 7.1 that the next activity that will be executed is A4.

For **Execution Phase**, the flow was started by Activity E4, in which all material is compiled and checked again to be sent to the participant. Next, E5 is performed, in which the environment, e-mails, and recordings were prepared to be performed on the day of the study. After the preparation of the evaluation and before the beginning of the execution, Activity E1 was performed, which is intended to send the Profile Questionnaire to the twelve (12) participants. After receiving the e-mail with the responses from the participants, the flow is executed again in the analysis phase, the activity to be executed being completed, the flow returns to Activity E2 in which the Informed Consent Term is presented and signed by the participants. In Activity E8, the Process Executor presents the DSLs introduction video, the tasks to be performed and the questionnaire that must be answered by the participants at the end of the evaluation. In Activity E9 it was the moment when the participants perform the tasks proposed by the Process Executor, the initial task was to configure the environment that they should perform the usage scenarios for each of the DSLs and at the end answer the Usability Questionnaire V2 (Appendix F). Once the previous activity is finished, the execution proceeds to Activity E7, in which the task data and the Usability Questionnaire V2 are sent by the participant by e-mail to the Process Executor. In Activity E10, the data collected during the execution of the evaluation are stored in the online database.

The **Analysis Phase** Workflow starts with Activity A4, in which the protocol analysis was performed, first by Process Executor (DSL developer) and later by another researcher experienced in the process, after the execution of this activity the flow returns to Execution Phase, that when executing some activities and returns again to Analysis Phase. At this

moment, Activity A1 is executed, which analyzes the profile data of participants received by email, after the analysis, the execution blocks are selected, that is, the separation of participants into homogeneous groups to prevent the sample from being invalid and avoid bias in the results, ending this activity the flow returns to process execution. At the end of all execution activities we went back to analyzing the collected data and ran Activity A7, in which the Process Executor analyzes the data collected through the Usability Questionnaire V2 (Appendix F), from the runtime metrics of each task and number of tasks completed. Finally, Activity A9 was executed, this activity consists of the analysis of the tasks and verification of consistency with the baseline.

In the last Phase, the **Report Phase** concentrates all the activities of presentation of the evaluation protocol that was carried out, this phase will be briefly reported in Section 7.1.1. Activity R5 presents the evaluation conduction, R4 explains the study protocol used in this quasi-experiment case (Usa-DSL brings a protocol template for this study and can be accessed at the process link <http://lesse.com.br/usa-dsl/>). Activity R1 consists of presenting the profile data of the participants, which were collected through the Profile Questionnaire (Appendix H). In R2 the number of participants is described and in R8 the instruments used, in this case, Informed Consent Term, Profile Questionnaire, Videos with the presentation of the DSL, documents with the description of the Tasks to be performed and the Usability Questionnaire. In Activity R9, the activities and sub-activities that the participants performed are detailed within the protocol. In R7, the data collected through the questionnaires were presented and discussed and, finally, in R11, the information obtained during the evaluation was discussed and finally compiled and revised.

7.1.1 Usability Test Definition

In this section the use of the Usa-DSL Process to evaluate a DSL that has an approach intended to be used by the industry will be presented. The Teasy Framework is composed of the language and the tool for generating sequences and automated functional test scripts. The purpose of this usability evaluation is to raise evidence on the ease of use and effort when performing web functional tests using the Teasy DSL. For this evaluation, qualitative metrics such as usability were raised, as well as quantitative metrics such as the effort (time spent) to run a set of predefined tasks. The technologies and tools used in the study were:

Robot Framework is an open source technology with the objective of performing the creation of system acceptance tests. Its use is based on an easy-to-learn tabular syntax [15]. The framework also uses natural language for the definition of the keywords, which are like test methods, and the Steps, which are the steps within each keyword.

Selenium WebDriver was used together with the Robot Framework as it is a technology responsible for handling browsers through interactions with the web application's graphical interface, representing the user's actions [3]. The technology captures the graphic elements through the attributes present inside the HTML, which in turn represent the user interface, enabling the carrying out of manipulations and actions with these elements.

Teasy Framework is composed by the language, structure and code generator. **Teasy Language:** is a language for the domain of functional testing in web applications. Teasy by definition is a projection language, which tends to reduce the necessary implementation time, providing projections based on the information added in its source code. **Teasy Structure:** will be used at the end of the implementation of the tests using the standard Page Objects in Teasy language and the code will be translated into executable test scripts. Imports and communication among files will be done through Teasy Structure and the transpiled files will be added in their respective directories. **Teasy Generator:** it is used in the complete functioning of the framework, as input and output data are presented through Teasy Generator (interface).

System Under Testing: during the study, two web applications were used as SUT: Shopping Cart Software during the evaluation, and ReactJSCalculator during the training stage. ReactJSCalculator¹ is a low-complexity web application that performs simple math operations. Shopping Cart Software² is an e-commerce that allows someone to register new users and edit their respective data. This application has all the components of applications existing in the industry, but during the evaluation, a demo version was used. The application makes it possible to purchase a product, edit the purchase, check the cart, search for new products, among other features.

For data collection, the following instruments were used: videos and tasks created by the DSL developer, and terms and questionnaires from the Usa-DSL Process. **Configuration Forms:** were filled out during the configuration of the environments, and had the objective of collecting the time spent with the configuration of each framework; **Feedback Forms:** These artifacts were created based on the Usability Questionnaire adapted from the "A Cognitive Dimensions (CD) Questionnaire" [17] and were filled in at the end of the evaluation run; **Source Codes and Time Spent:** at the end of the execution, each participant provided the source code created, together with the time spent in each activity.

The execution of this study followed the Phases, Steps, and Activities of the Usa-DSL Process.

The evaluation was carried out in January 2021 and consisted of two stages, these being training and the supervised execution of the quasi-experiment. For this study twelve (12) participants with different levels of experience in software development and testing were invited. To conduct the evaluation, in the Planning Phase, video training was developed

¹ReactJSCalculator: <https://github.com/yuryalencar/ReactJSCalculator>

²Shopping Cart Software: <http://demo.cs-cart.com/>

for each of the frameworks used. The training consisted of setting up the environment and instructions for using both frameworks. Due to the pandemic, it was not possible to carry out the study face to face, for this reason, each participant was individually invited to carry out a profile survey through a video call. The profile information collected was used to perform the blocks based on the participants' experiences in functional tests. The distribution of participants was performed randomly and took into account the level of experience with software testing.

On the day scheduled for the evaluation, the (12) twelve participants entered a video call so that the execution had control over the start and end time. At this time, the tasks that should be performed using the Shopping Cart Software application were made available. The tasks were made available in a document shared among everyone. **Activity 1:** searches for a specific product within the application and check its existence; **Activity 2:** simulates registration in the application, add a product to the cart, and continue shopping; **Activity 3:** simulates a registration in the application, add a product to the cart, and complete the checkout; **Activity 4:** creates a sequence using Teasy Generator and check the amount of generated tests. This activity was only requested from participants who used the Teasy Framework due to framework compatibility. At the end of the tasks execution, each participant answered the Usability Questionnaire V2 (see Appendix E), about the use of the frameworks. This form consisted of questions related to usage and functionality, as well as feedback and suggestions for improvement. The analysis of the results was realized by looking at the completed tasks, the execution time, and the answers of the Usability Questionnaire. For this work we will focus on the analysis of the questions answered in the questionnaire, as we are checking the opinion of the participants regarding the usability of Teasy.

Regarding the question "Does the Teasy Framework have a good usability and representativeness with the domain?", 66.7% of the participants answered that the Teasy DSL development environment does not present any problems, that is, the language syntax is clear, and each file has its role well defined, not allowing the user to make mistakes easily. Participants agree that maintainability is easy and 83.3% of them responded that the automatic update of dependencies and the final code of execution are easy to read and understand, and the reuse allows for an easy maintenance of the created tests. As for the ease of use of the Teasy Generator 66.7% of the participants mentioned that it is simple to generate new test sequences with its help, and responded that all or most of the generated sequences were representative for the SUT. As for the representativeness of the domain, 66.7% of the participants responded that they would use the Teasy Framework again due to its practicality, quality of the generated code, reuse, and maintainability. However, 16.7% answered that they would not use it because they know other technologies and have greater familiarity, the rest of the participants did not justify why they would not use the solution. Among the benefits of using Teasy, the following were highlighted: reuse of components, effort reduction, clear and well-defined syntax, complete report and does not require prior

programming knowledge. Regarding limitations were highlighted: complicated installation, lack of documentation, manual configuration.

7.2 Study 2: Heuristic Evaluation

In this section, the execution of Study 2 performed to evaluate the Teasy DSL will be presented. First, we will present the workflow followed, the activities performed and the artifacts used, in order to understand how the Process Executor conducted the evaluation using the Usa-DSL Process. In order not to repeat the data presented above, we will only highlight the activities that undergo changes in the Heuristic Evaluation workflow.

Life Cycle: This evaluation followed the life cycle presented in Figure 7.2 and follows partially the description presented in Section 7.1.

Execution Flow: the flow that was followed to execute the Heuristic Evaluation that has a change in the Planning Phase workflow, as it does not execute Activity P6 because it does not work with metrics, but with heuristics. Thus, changing the type of evaluation instrument and data analysis, the other activities follow the flow presented in Section 7.1. Another difference in this type of evaluation is the protocol that is followed, as it uses the Survey method, which is carried out completely online by the participant sending the results of the evaluation by email at any time they wish.

7.2.1 Heuristic Evaluation Definition

In order to verify the Usa-DSL Process in practice, an inspection evaluation of the usability of the Teasy Language language was carried out. For this study, a Heuristic Evaluation was carried out using the empirical Survey method. Although Teasy Language is a projectional DSL and makes use only of texts, we consider that an inspection evaluation would help to understand and find problems in the interface and in the representativeness of the text elements in relation to the domain. Thus, for the evaluation to be performed, the Heuristic Evaluation Checklist for Textual DSL [64] was used. That said, the motivation of this study was to analyze the ease of use of the Teasy Language, as we believe that its usability can directly impact its use by testers. Thus, this study aims to answer the following questions: Is Teasy Language easy to use? Are your elements, highlights and feedbacks representative? How severe are the existing usability issues? If there are usability issues, is it possible to use DSL, or is an urgent update needed?

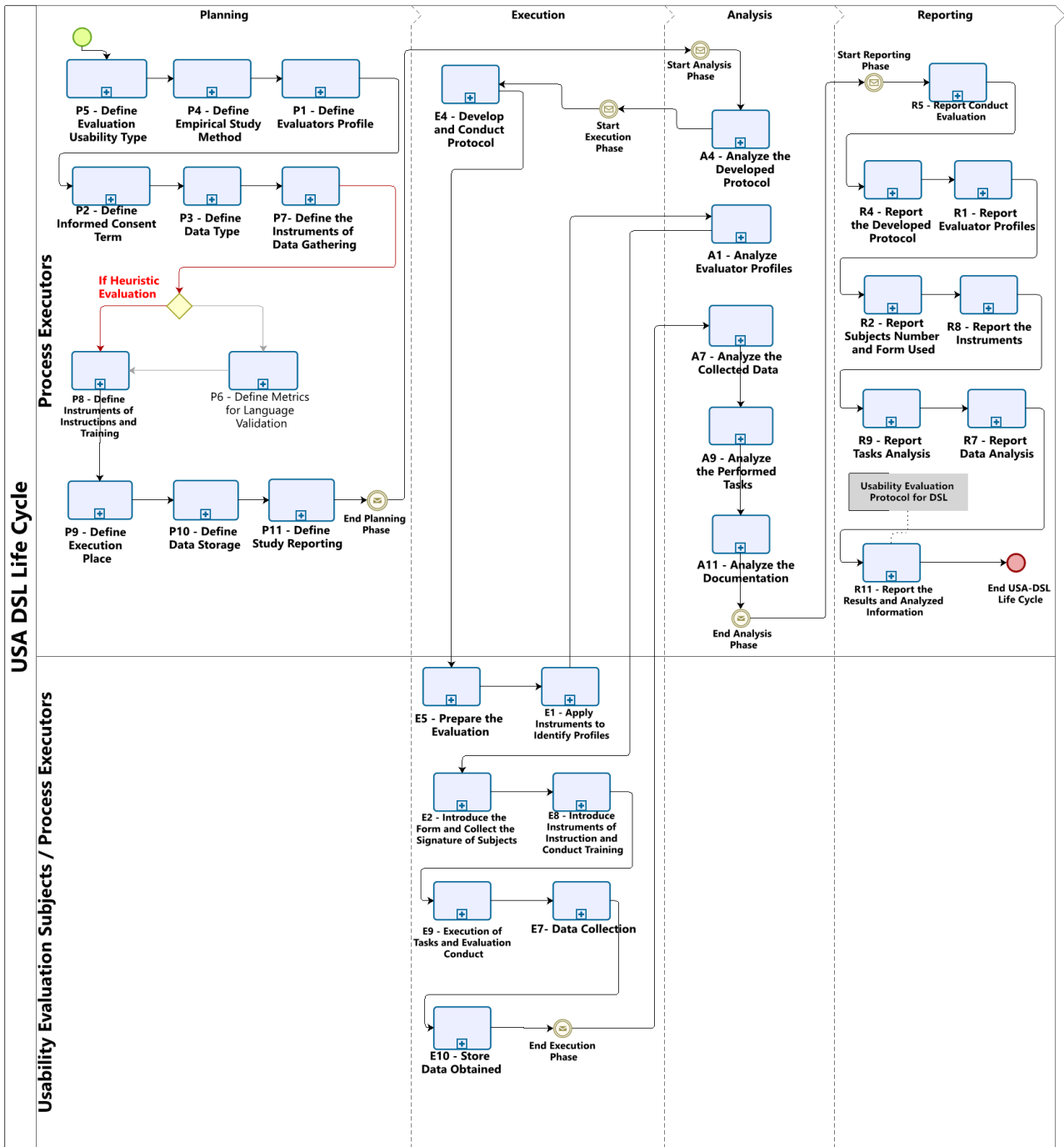


Figure 7.2: Heuristic Evaluation Workflow

For data collection, the following instruments were used: a video explaining the use of the Teasy Framework³; specification of activities to be performed⁴; Informed Repair Term; Profile Questionnaire and Heuristic Evaluation Checklist for Textual DSL. During the instrument preparation stage, we took special care regarding the participants' contact with Teasy Language. Thus, we use the part of the training related to the Teasy Framework that presents how to use the language. Videos that explained the configuration and use of the

³Teasy Framework Training: <https://youtube.com/playlist?list=PLrua1dSU7PC0fHVkqZqZWFefDZC57t7wy>

⁴Specification of tests used: <https://url.gratis/giwi3>

Teasy language were sent to the participants. As a final result of the training an executable code was using the Teasy Language.

The study was conducted online, in January 2021. Four (4) participants were invited for the evaluation and contacted via email. The sample was selected by convenience and by indication, as we were looking for professionals with experience in Human-Computer Interaction (HCI), specifically in heuristic evaluation. After acceptance, the survey execution instruments were sent to the participants by e-mail.

To preserve the identity of the participants, the profile questionnaire responses, as well as the other study responses will be identified by the label assigned to each participant, that is, **E1**, **E2**, **E3**, and **E4**, the profile surveyed for each participant will be described below.

E1: The expert has used a DSL before, has also participated in empirical studies such as case studies, controlled experiments and *Surveys*, is currently an undergraduate student and has 1 to 3 years of industry experience. The subject also has one year of experience with functional software tests, as well as experience in heuristic evaluations.

E2: The expert has never used a DSL before, but has participated in empirical studies such as case studies, proofs of concept and *Surveys*, is currently an undergraduate student and has 1 to 3 years of experience in the industry. The subject does not have experience with functional software tests, but has experience in heuristic evaluations.

E3: The expert has both used a DSL and created one before, participated in empirical studies such as controlled experiments, case studies, quasi-experiments, proofs of concept and *Surveys*, is currently an undergraduate student and has 1 to 3 years of industry experience. The subject also has 1 to 3 years of experience with functional software tests, in addition to experience in heuristic evaluations.

E4: The specialist has used a DSL before, has also participated in empirical studies such as case studies, controlled experiments, quasi-experiments and *Surveys*, is currently pursuing a PhD and has no experience in the industry. The subject has up to one year of experience with functional software tests, in addition to experience in heuristic evaluations.

After collecting the answers from the four participants, a qualitative analysis of the results obtained through the Heuristic Checklist was performed. Each of the problems found and its degree of severity (according to the table presented in the Checklist) attributed by the participants were analyzed.

H1: Visibility of system status - no expert found any problems related to the ambiguity of the existing commands in the language, that is, each command specifically performs an action. All users stated that the language was correctly coupled, when an update of some information in a class is needed, this information is reflected in all places that use it. Finally, none of the subjects identified problems in the Teasy Language alert messages, so it can be seen that the language provides messages on the screen long enough to be read.

The **E4** expert detected that the language does not immediately inform when the file is saved, this same guy classified this error as 1, a cosmetic usability problem.

H2: Match between language and the real world domain - all experts agreed that Teasy Language has elements that meet the domain of functional software testing and that all these elements have a good representation. The **E4** expert mentioned that according to his experience a specific keyword may not be coherent with the domain. However, this problem was rated 1, that is, a cosmetic error that does not affect the use of Teasy Language.

H3: User control and freedom - all experts have stated that the language has redundancy in the undo and redo commands, allowing the user a freedom to restore the code state at any time during development.

H4: Consistency and standards - all textual elements present in the language are connected to each other according to their names and using the same style. About shortcuts, one of the experts showed that in a few moments free text is allowed and that he does not consider this a usability error, this point was expected because the language is projectional, most of the code is generated for the tester. Between the components connection the **E4** expert found inconsistencies in the highlight, this error was classified as 2, therefore, a small usability error and has low correction priority. Another error detected by the same participant is that the error highlighting mechanism takes time to appear, also classified as a small usability error.

H5: Error prevention - all experts have confirmed that the language predicts wrong decisions by users and warns them. With regard to requesting some confirmation, when a user performs actions considered dangerous, the experts were unable to verify due to the scenarios used, which do not require confirmation. The expert **E4** had problems when trying to delete a typo using the "DELETE" key, instead of deleting only one character, he deleted a component, the expert classified this error as 1, that is, only one cosmetic problem.

H6: Recognition rather than recall - all participants confirmed that Teasy Language has keywords that are easy to remember and meaningful to the software testing domain. In addition, experts also did not find usability problems in highlights, that is, the highlights increased the readability of the code created through the language.

H7: Flexibility and efficiency of use - all experts have confirmed that the language offers all the desired components to complete the tasks. Because the Teasy language is projectional, it does not have shortcuts for experienced users, all users receive the same projection, even so the **E3** expert classified this absence as a usability problem, however he classified it as a small error and that does not affect usage efficiency. The subject **E4** detected that the development environment has some limitations, such as it is not possible to copy an element using the mouse, requiring the use of the keyboard and it is also not possible to delete a component using the backspace key, when the focus was on the beginning of some placeholder, both errors were categorized as 3, that is, serious errors that should be prioritized in a future version.

H8: Aesthetic and minimalist design - all experts agree that the information needed to understand a component or information is presented in short texts, no longer than three lines. In some components, the **E4** specialist detected some distinct additional information highlights with non-standard colors, considered as a small error, of low priority.

H9: Help users recognize, diagnose, and recover from errors - all experts answered that in case of an error, the DSL shows an appropriate error message, where after reading the user knows how to recover from the error or where to look for a solution. In case of problems, the messages present the possible problems clearly and objectively. In addition, the Teasy language has a mechanism to highlight syntax errors in text or reserved words, the Teasy language also has mechanisms to highlight errors in nesting or organization of reserved words that can make the model inconsistent. The **E3** specialist could not detect in the Teasy language mechanisms that mark connection errors between elements that can make the model inconsistent, this error was classified as cosmetic only.

H10: Help and documentation - all experts confirmed that the Teasy language has contextual help for each element. The experts **E4** and **E3** detected problems related to the tutorial, both showed that there is no textual tutorial classifying it as a small error of low priority. Experts **E4** and **E3** also found problems in the support provided by the commands, both claimed that the help texts are too small and have some grammar errors, classifying it as 2, a small usability problem. In summary, experts detected some usability problems in Teasy Language, however, they did not directly impact the functioning and understanding of the language. Thus, there is evidence that even in the first version of Teasy Language, participants understood the purpose of the language and were able to use it. Finally, all errors regarding the editor and the language will be resolved in a future version, thus the Heuristic Evaluation contemplated all the objectives specified for the study.

Regarding the Heuristic Checklist, expert E4 mentioned that the use of questions within the heuristics mischaracterizes the evaluation technique a little, making it similar to applying a checklist instead of being an inspection. I felt that my power to express errors was limited when I needed to fit a problem into an issue and not into the heuristic as a whole. Maybe this must have happened because we use the Google form format that does not give visibility to the whole, but in the next evaluations, we will try to give more freedom to the specialist.

7.3 Process Executor Opinion - Usa-DSL Process

This section will present the perception of the Process Executor when using the Usa-DSL Process, for conduct the Study 1 and Study2, for this we use an online questionnaire, in which we obtain the Perceived Usefulness (PU) and Ease of Use (EoU) using the Technology

Acceptance Model (TAM) [23]. We chose this model because the main goal of this study was to obtain the respondents' perception about the process content, presentation and execution.

When starting the evaluation, we asked the respondent to read the Informed Consent Term and if he agreed with the terms, he signed to continue the evaluation. Afterward, we identified your profile through four questions: Name of Institution or Company: Alirok; Scroll that best describes your job at institution or company: Tech Lead; Time of experience using or designing DSL (in years): 3 years and Country: Brazil. After answering about his profile, he moved on to the Perceived Usefulness (PU) and Perceived Ease of Use (PEoU) questions about the use of the process, these questions followed a 5-point Likert scale (Strongly Disagree, Disagree, Neutral, Agree and Strongly Agree). At the end of the questionnaire, three open questions were answered so that the participant could freely discuss the use, adoption and recommendation of the process.

As a result, in the questions about Perceived Usefulness (PU) 5 out of 6 were answered with Strongly Agree, only one Agree.

Strong Agree answers were from questions: PU1. The documentation provided by the Usa-DSL Process helps me understand what should be done; PU2. The Usa-DSL Process helps me understand how to perform a DSL usability evaluation; PU3. The elements that make up the Usa-DSL Process help me understand what the Process proposes; PU4. The Usa-DSL Process improves the quality of planning, execution, analysis and report with regard to DSL usability evaluation and PU6. Overall, I find the Usa-DSL Process is useful in performing usability evaluation for DSL.

The only one with an Agree answer was from question PU5. Although the Usa-DSL process is long, its benefits outweigh. In general terms, the participant responded that the usability of the process is very good, despite considering the process long, he believes that the benefits outweigh this problem. In the questions about Perceived Ease of Use (PEoU), the results obtained 2 Neutral, 3 Agree and 2 Strong Agree.

Neutral responses were from the questions: PEoU1. The Usa-DSL Process is objective. I often do not become confused when using it and PEoU2. Usa-DSL Process is easy to use for fully document elements of a usability evaluation for DSL.

The **Agree** answers refer to the questions: PEoU3. The process flow is easy to use when performing a usability evaluation for DSL; PEoU5. Browsing Usa-DSL Process pages is easy and PEoU7. Interaction with the process requires a little of my mental effort. **Strong Agree** answers correspond to questions: PEoU4. The process has a lot of information, but it is easy to plan, execute, analyze and report a usability evaluation for DSL using it and PEoU6. It is easy to find the necessary elements to perform usability evaluation for DSL in the Usa-DSL Process.

In summary, the participant did not respond positively about the ease of use of Usa-DSL, and does not consider Usa-DSL objective, becoming confused when using it. But

he agrees that the process flow and navigation on his pages are easy to use, but interacting with the process takes a little mental effort. The participant strongly agrees that it is easy to plan, execute, analyze and report a usability evaluation for DSL using Usa-DSL. And even if the process has a lot of information, it is easy to find the necessary elements to carry out the evaluation in the Usa-DSL process.

As for open questions, OP1. What is your opinion/statement on the Usa-DSL Process for developing and conducting a usability evaluation for DSL? The participant just replied that the documentation is easy to use. In PO2. Would you adopt for Usa-DSL Process evaluating the usability of a DSL? He replied that he had already adopted/used the process and that it was very easy to document his evaluation. Already in PO3. Would you recommend a USA-DSL Process colleague to support performing usability evaluation for DSL? The participant responded that they would recommend Usa-DSL to their peers, because the process has extensive documentation and is customizable to their needs.

7.4 Final Considerations

Although these evaluations (Study 1 and Study 2) were the first complete use of the Usa-DSL Process, we noticed that the process was able to help the Teasy developer to evaluate the usability of his DSL. It seems that Usa-DSL achieved its objective, which was to instruct the evaluator regarding the steps to be taken during all stages of the evaluation. We can highlight as one of the benefits of using the process the use of the questionnaires "Usability Questionnaire" and "Heuristic Evaluation Checklist for Textual DSL", as they helped to identify important points about DSL usability.

As for the results of the evaluations (Study 1 Section 7.1 and Study 2 Section 7.2), the developer of Teasy Language was able to infer that in its first version it already has good usability, for example, it has several features that allow the tester to locate himself in the code, as well as understand and solve possible problems. Another information raised by the developer was the possibility of detecting problems and understanding the severity and urgency that must be resolved in its next version, thus increasing the quality of the language.

In response to the evaluation of the use of the Usa-DSL Process (Process Executor Opinion - Section 7.3), and considering the issues raised by the participant, we highlight that the process was a way we found to document the Usa-DSL Framework and explain the execution flows of usability evaluation for developers, DSL analyst, and testers, were able to autonomously conduct their usability evaluation. We are aware of the limitations of the process and as a way to mitigate them, we are developing a tool to support the Usa-DSL Process and automatically develop the usability evaluation provided for in the process, with the support of all the work products it has.

8. THESIS SUMMARY AND FUTURE WORK

Domain-Specific Languages (DSLs) can be used by software architects or developers to model or code the desired application. However, DSLs do not always represent their domain directly, which can be responsible for their failure. One of the reasons is due to the way DSL designers generally assess the usability of their language, *i.e.* using *ad hoc* processes. This could be the result of a lack of experience in usability assessment. Even though some work has taken usability, user experience and design principles into account when developing DSL. Understanding these principles when designing and evaluating a DSL is challenging, as not all design goals have the same relevance across different systems or DSL. Furthermore, existing approaches are not well defined and may not bring the results expected by such designers. Therefore, it is important that DSL designers use a well-defined usability assessment process to facilitate the conduct of proper assessments to assess the correct perception of their DSL users, *i.e.*, architects, developers, or end users.

In this context, this thesis contributed with theoretical and empirical studies to advance in the areas of Software Engineering and Human-Computer Interaction using well-defined techniques to assess the usability of DSLs. The results provided evidence for a positive answer to our research question proposed in Chapter 1, *i.e.* “*Can a usability evaluation process for DSL help Language Engineers to plan, execute, analyze, and report their evaluation?*”. We have indication that the Usa-DSL Process was able to help language engineers to Plan, Execute, Analyze and Report their evaluations. This evidence can be seen in Chapter 7. That chapter presented the Usa-DSL process being executed by a language engineer and showed how he followed/produced each of the steps, phases, activities, tasks and work products.

Section 8.1 revisits our research questions and discusses the thesis contributions. Section 8.2 summarizes the limitations of the study and also outlines ongoing research and future work. Finally, Section 8.3 describes the author’s academic contribution in terms of publications.

8.1 Thesis Contributions

As already stated, the contributions of this thesis can be summarized as:

Usa-DSL Process - Usability Evaluation Process for Domain-Specific Languages: We proposed, design and develop the Usa-DSL Process (Chapter 4). The Usa-DSL seeks to assist Language Engineers to evaluate their languages regarding ease and quality of use, with no need to have deep knowledge on usability approaches. Usa-DSL is an iterative usability evaluation process [82] and is intended to help the process performer answer

questions “Who will do what, when, and how?” [76] when using the process. This process was developed based on the Usa-DSL Framework [62], which is composed of *Phases*, *Steps*, and *Activities*. Its structure is composed of Method Content and Process [58]. The Method Content (Section 4.5) is organized into Work Products, Profiles, Tasks, and Steps. The Process (Section 4.2) has the Phases, Activities, Profile Use, Task Use, and Work Product Use elements, as well as groups of elements that act as guidelines, *i.e.*, templates of documents (Informed Consent Term, Heuristic Checklist, Usability Questionnaire) that will be consumed, generated, or modified throughout the evaluation life cycle.

Usa-DSL Process Modeling: We modeled Usa-DSL using SPEM [58] and BPMN [59] (Chapter 4). SPEM was adopted to describe most of the process (see Section 4.2) and BPMN to represent the life cycle flows of our process (see Section 4.4). Even though SPEM provides a way to represent flows through UML diagrams, we decided to use BPMN, since it provides a straightforward way to represent sub-activity flows for each phase and the possibility to represent task flows as sub-processes of each of its activities. To describe the Usa-DSL Process life cycle, as mentioned before, we have used BPMN. This modeling was performed in three (3) levels: the first level describes the interactions among the process Phases; the second level describes the interaction among Activities; and, the third level presents the interactions among process Tasks. The process was developed using the Eclipse Process Framework - EPF Composer, which helped in the structuring, documentation, and publication of the process.

Evaluation Process: To first introduce the Usa-DSL process to the community, we performed a qualitative study based on a survey with 21 experienced participants who freely used and explored our process to provide feedback on Perceived Usefulness (PU) and Ease of Use (EoU) dimensions using the Technology Acceptance Model (TAM) [23] (Chapter 5). We chose this model because the main goal of this study was to obtain the respondents perception about the process content and presentation. In relation to the general evaluation analysis, we understand that most respondents said they would use and indicate the process to their colleagues and other researchers. Nonetheless, a small number of respondents believe that the process should undergo enhancements to be used in all contexts, such as to evaluate simpler DSLs. In this context, we believe that with the development of the process supporting tool, this suggestion will be addressed soon.

Artifact Design and Evaluation: We developed a structured way to build a Heuristic Evaluation Checklist (HEC) for DSLs (Chapter 6). This checklist is different from traditional checklists since it is focused on DSL. Once a checklist is provided, the evaluators only follow a set of heuristics and freely point out the errors found when using the DSL. Basically, the produced checklist provides a set of questions, based on the heuristics that direct an evaluation for a specific domain (Appendix D). The methodology to develop our Heuristic Evaluation Checklist follows the methodology proposed by [69] (Section 4.6.1). The first steps were: to understand Heuristic Evaluation and Domain-Specific Language concepts, to adapt existing

heuristics for the DSL domain, to produce a set of questions based on a systematic literature review [70], and to create our initial Heuristic Evaluation Checklist. After that, we submitted this preliminary checklist to be evaluated by a set of HCI experts through interviews. Finally, we got a modified Heuristic Evaluation Checklist that we applied to an example of use.

Usa-DSL Process - Example of Use: After completion of the Usa-DSL Process, two evaluations were conducted. These evaluations were aimed at evaluating the Teasy Framework which is composed of its DSL, sequence generation tool, and automated functional test scripts. It should be noted that the DSL designer used the Usa-DSL Process to conduct the two evaluations, that is, the process was put into practice. In order to identify each of the usability evaluations conducted, we called the first Study 1 and the second Study 2. Study 1 used the usability evaluation method "Usability Test", which was performed using the empirical method "Quasi-experiment". Study 2 used the usability evaluation method "Heuristic Evaluation", which was performed using the empirical method "Survey". From the analysis of the results of the evaluations, the language designer can highlight the usability problems that the DSL presented and carry out the suggested changes. At the end of the evaluations, we asked the DSL designer to answer a questionnaire to get their perception about the use of the process. As a result, the designer responded that the usability of the process is very good and despite considering it long, he believes that the benefits outweigh this problem. When it comes to ease of use, he does not consider the Usa-DSL process objective and sometimes gets confused when using it. But he agrees that the flow of the process and navigating its pages are easy, but interacting with the process takes a bit of mental effort. The designer strongly agrees that it is easy to plan, execute, analyze and report a usability assessment for DSL using Usa-DSL. And even if the process has a lot of information, it is easy to find the necessary elements to carry out the assessment in the Usa-DSL process.

8.2 Limitation and Future Work

Despite this thesis having presented real contributions to the Usability Evaluation for DSL, Usability Evaluation Process Development, and the development and adaptation of artifacts of usability, we identified the limitations of the thesis contributions that can be dealt with in the future. It is worth remembering that previous chapters have already discussed specific limitations. Hence, we describe in this section broader limitations to be overcome as well as pieces of work that are in development and opportunities for future work made during short and medium-term research.

Limitation of Usa-DSL Process: Participants in the Survey Study (Chapter 5) believed that the process is extensive and that for someone who does not have prior knowledge regarding usability evaluation it still requires some time to be executed. Specifically, about the perception Massiveness, we emphasize that the process is executed in flows and that the

Process Executors do not need to go through all activities to evaluate its DSL. Instead, the Process Executors choose activities that correspond to the usability evaluation that they want to perform. It is worth noting that the process was designed to be used for users who have no knowledge on usability evaluation. Therefore, it is highly recommended that all process elements store the necessary instructions, step-by-step, on how the user executes them.

Ongoing Work - Usa-DSL Tool: in order to speed up the process of building the needed artifacts described in our process, we are implementing a tool to help DSL designers to build their usability evaluation. The tool, called Usa-DSL Tool, will generate all artifacts, based on templates, for example, invite e-mails, questionnaires, informed consent term, and so on. It is a collaborative web tool, which aims to guide the user in the execution of the Usa-DSL process in domain-specific language usability evaluations. Performing the process intuitively and iteratively through all its phases. To develop this tool we are using the PHP 7.4 programming language, the Laravel 5 framework and for the front end HTML 5, CSS3, JavaScript, Bootstrap. An example of part of this tool is shown in Figure 8.1. This research is being co-supervised by the author of this thesis.

The image shows two screenshots of the Usa-DSL Tool interface. The top screenshot displays the 'Evaluation and Study Types' step (Step 1). It features a sidebar with navigation options: PROJECT (My Projects, New Project), ACCOUNT SETTINGS (Profile), and a 'NEXT' button. The main content area has the title 'Evaluation and Study Types' and two sections: 'Select the usability evaluation type' with radio buttons for 'Usability Evaluation' (selected) and 'Heuristic Evaluation', and 'Select the empirical method study' with radio buttons for 'Survey', 'Case Study', 'Experiment', and 'Quasi-Experiment'. The bottom screenshot displays the 'Study Information' step (Step 2). It features the same sidebar and a 'PREV' button. The main content area has the title 'Study Information' and several input fields: 'Responsible researcher', 'Language to be evaluated', 'Evaluation objective', 'Date to start' and 'Date to end' (with date pickers), and 'Institution or company'. A 'NEXT' button is located at the bottom right.

Figure 8.1: Usa-DSL Tool (Interface Sample)

Future Work - Usa-DSL Evaluation: The Usa-DSL Process is being used to evaluate the usability of the DSL PerfText, a new text representation version of Canopus [14], which is a textual language for the performance testing domain. This evaluation is in the planning stage, but the DSL designer has been sending us feedback related to the use of Usa-DSL and mentions that the process is easy and does not need help to be carried out. As the usability evaluation to be performed uses the Usability Test method, the designer

analyzed the Usability Questionnaire template and informed us that it will be used partially, as its objective is to understand how much the reserved words of the language represents the domain.

8.3 Publications

During the development of this thesis we presented and discussed our research results in the following papers:

- Poltronieri, I.; Zorzo, A.F.; Bernardino, M.; Pedroso, A.; Oliveira Junior, E.A. **under review**, Usa-DSL: a Process for Usability Evaluation of Domain-Specific Languages. Submitted to Journal of Universal Computer Science, 2021.
- Poltronieri, I.; Pedroso, A. C.; Zorzo, A. F.; Bernardino, M.; de Borba Campos, M. “Is usability evaluation of DSL still a trending topic?” In: Human-Computer Interaction: Theory, Methods and Tools (HCII), 2021, pp. 299–317.
- Poltronieri, I.; Zorzo, A.; Bernardino, M.; Medeiros, B.; Campos, M. “Heuristic evaluation checklist for domain-specific languages”. In: International Conference on Human-Computer Interaction Theory and Applications (HUCAPP), 2020.
- Rodrigues, I. P.; Zorzo, A. F.; Bernardino, M. “Uma Proposta de Processo de Avaliação de Usabilidade para DSLs”. In: Anais da III Escola Regional de Engenharia de Software (ERES), 2019, pp. 127–136.

REFERENCES

- [1] Albuquerque, D.; Cafeo, B.; Garcia, A.; Barbosa, S.; Abrahão, S.; Ribeiro, A. “Quantifying usability of domain-specific languages: An empirical study on software maintenance”, *Journal of Systems and Software*, vol. 101, Mar 2015, pp. 245 – 259.
- [2] Alhaag, A. A.; Savic, G.; Milosavljevic, G.; Segedinac, M. T.; Filipovic, M. “Executable platform for managing customizable metadata of educational resources”, *The Electronic Library*, vol. 36–6, Dec 2018, pp. 962–978.
- [3] Avasarala, S. “Selenium WebDriver practical guide”. Birmingham, WM, UK: Packt Publishing Ltd, 2014, 1st ed.
- [4] Bacikova, M.; Galko, L.; Hvizdova, E. “Manual techniques for evaluating domain usability”. In: 14th International Scientific Conference on Informatics, 2017, pp. 24–30.
- [5] Bacikova, M.; Maricak, M.; Vancik, M. “Usability of a domain-specific language for a gesture-driven ide”. In: Federated Conference on Computer Science and Information Systems, 2015, pp. 909–914.
- [6] Barbosa, S. D. J.; Silva, B. S. d.; Silveira, M. S.; Gasparini, I.; Darin, T.; Barbosa, G. D. J. “Interação Humano-Computador e Experiência do Usuário”. Autopublicação, 2021.
- [7] Barisic, A.; Amaral, V.; Goulao, M.; Aguiar, A. “Introducing usability concerns early in the dsl development cycle: Flowsl experience report”. In: 1st International Workshop on Model-Driven Development Processes and Practices, 2014, pp. 10.
- [8] Barisic, A.; Amaral, V.; Goulão, M. “Usability evaluation of domain-specific languages”. In: 8th International Conference on the Quality of Information and Communications Technology, 2012, pp. 342–347.
- [9] Barisic, A.; Amaral, V.; Goulão, M.; Barroca, B. “Quality in use of domain-specific languages: A case study”. In: 3rd Workshop on Evaluation and Usability of Programming Languages and Tools, 2011, pp. 65–72.
- [10] Barisic, A.; Amaral, V.; Goulão, M.; Barroca, B. “Evaluating the usability of domain-specific languages”. In: *Formal and Practical Aspects of Domain-Specific Languages:Recent Developments*, IGI Global, 2012, chap. 14, pp. 386–407.
- [11] Barisic, A.; Cambeiro, J.; Amaral, V.; Goulão, M.; Mota, T. “Leveraging teenagers feedback in the development of a domain-specific language: The case of programming low-cost robots”. In: 33rd Symposium On Applied Computing, 2018, pp. 1221–1229.

- [12] Barisic, A.; Goulão, M.; Amaral, V. “Usability Driven DSL development with USE-ME”. In: *Computer Languages, Systems and Structures*, 2018, pp. 118 – 157.
- [13] Barnum, C. “Usability Testing Essentials: Ready, Set ...Test!” San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2020, 2th ed.
- [14] Bernardino, M.; Zorzo, A.; Rodrigues, E. “Canopus: A Domain-Specific Language for Modeling Performance Testing”. In: *9th International Conference on Software Testing, Verification and Validation*, 2016, pp. 157–167.
- [15] Bisht, S. “Robot framework test automation”. Birmingham, WM, UK: Packt Publishing Ltd, 2013.
- [16] Blackwell, A.; Green, T. “A cognitive dimensions questionnaire”. Source: <http://www.cl.cam.ac.uk/~afb21/CognitiveDimensions/CDquestionnaire.pdf>, Mar, 2019.
- [17] Blackwell, A.; Green, T. “A cognitive dimensions questionnaire optimised for users”. In: *12th Annual Workshop of the Psychology of Programming Interest Group*, 2000, pp. 10–13.
- [18] Braun, V.; Clarke, V. “Using thematic analysis in psychology”, *Qualitative Research in Psychology*, vol. 3–2, Dec 2006, pp. 77–101.
- [19] Cachero, C.; Melia, S.; Hermida, J. M. “Impact of model notations on the productivity of domain modelling: An empirical study”, *Information and Software Technology*, vol. 108, Apr 2019, pp. 78 – 87.
- [20] Clark, L.; Pantidi, N.; Cooney, O.; Doyle, P.; Garaialde, D.; Edwards, J.; Spillane, B.; Gilmartin, E.; Murad, C.; Munteanu, C.; Wade, V.; Cowan, B. R. “What makes a good conversation?: Challenges in designing truly conversational agents”. In: *Conference on Human Factors in Computing Systems*, 2019, pp. 475:1–475:12.
- [21] Cronbach, L. “Coefficient alpha and the internal structure of tests”. In: *Psychometrika*, Elsevier, Sep, 1951, vol. 16, pp. 297–334.
- [22] Cuenca, F.; Bergh, J. V. d.; Luyten, K.; Coninx, K. “A user study for comparing the programming efficiency of modifying executable multimodal interaction descriptions: A domain-specific language versus equivalent event-callback code”. In: *6th Workshop on Evaluation and Usability of Programming Languages and Tools*, 2015, pp. 31–38.
- [23] Davis, F. “Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology”, *MIS Quarterly*, vol. 13–3, Sep 1989, pp. 319–340.
- [24] Eclipse Foundation. “Eclipse Process Framework Project - EPF Composer”. Source: <https://projects.eclipse.org/projects/technology.epf>, Nov, 2018.

- [25] Ewais, A. B.; De Troyer, O. "A usability evaluation of graphical modelling languages for authoring adaptive 3d virtual learning environments". In: 6th International Conference on Computer Supported Education, 2014, pp. 459–466.
- [26] Fowler, M. "Language workbenches: The killer-app for domain specific languages?" Source: <http://www.martinfowler.com/articles/languageWorkbench.html>, Aug, 2018.
- [27] Fowler, M. "Domain Specific Languages". Hoboken, NJ, USA: Addison-Wesley Professional, 2010, 1st ed.
- [28] Gibbs, I.; Dascalu, S.; Harris, F. C.; Jr. "A separation-based UI architecture with a DSL for role specialization", *Journal of Systems and Software*, vol. 101, Mar 2015, pp. 69 – 85.
- [29] Gilson, F. "Teaching software language engineering and usability through students peer reviews". In: 21st International Conference on Model Driven Engineering Languages and Systems: Companion, 2018, pp. 98–105.
- [30] Green, T.; Blackwell, A. "Cognitive dimensions of information artefacts: a tutorial. version 1.2". Source: <http://www.ndirect.co.uk/~thomas.green/workStuff/Papers/>, Mar, 2019.
- [31] Green, T. R.; Petre, M. "Usability analysis of visual programming environments: A cognitive dimensions framework", *Journal of Visual Languages & Computing*, vol. 7–2, Jun 1996, pp. 131 – 174.
- [32] Green, T. R. G. "Cognitive dimensions of notations". In: Conference of the British Computer Society Human Computer Interaction Specialist Group - People and Computers V, 1989, pp. 443–460.
- [33] Henriques, H.; Lourenço, H.; Amaral, V.; Goulão, M. "Improving the developer experience with a low-code process modelling language". In: 21th International Conference on Model Driven Engineering Languages and Systems, 2018, pp. 200–210.
- [34] Hermawati, S.; Lawson, G. "Establishing usability heuristics for heuristics evaluation in a specific domain: Is there a consensus?", *Applied Ergonomics*, vol. 56, Sep 2016, pp. 34 – 51.
- [35] Hesenius, M.; Gruhn, V. "Gesturecards-a hybrid gesture notation", *Human-Computer Interaction*, vol. 3, Jun 2019, pp. 1 – 35.
- [36] Hewett, T. T.; Baecker, R.; Card, S.; Carey, T.; Gasen, J.; Mantei, M.; Perlman, G.; Strong, G.; Verplank, W. "ACM SIGCHI curricula for human-computer interaction", Technical Report, New York, NY, USA, 1992, 173p.

- [37] Hoffmann, B.; Chalmers, K.; Urquhart, N.; Guckert, M. “Athos - a model driven approach to describe and solve optimisation problems: An application to the vehicle routing problem with time windows”. In: 4th International Workshop on Real World Domain Specific Languages, 2019, pp. 1 – 10.
- [38] Kabáč, M.; Volanschi, N.; Consel, C. “An evaluation of the diasuite toolset by professional developers: Learning cost and usability”. In: 6th Workshop on Evaluation and Usability of Programming Languages and Tools, 2015, pp. 9–16.
- [39] Kadoda, G.; Stone, R.; Diaper, D. “Desirable features of educational theorem provers - a cognitive dimensions viewpoint”. In: The Psychology of Programming Interest Group, 1999, pp. 1–6.
- [40] Kitchenham, B. “Procedures for performing systematic reviews”, Technical Report, Keele University, Australia, 2004, 26p.
- [41] Le Moulec, G.; Blouin, A.; Gouranton, V.; Arnaldi, B. “Automatic production of end user documentation for DSLs”, *Computer Languages, Systems and Structures*, vol. 54, Dec 2018, pp. 337 – 357.
- [42] Lima, Y. A. “Teasy framework: uma solução para testes automatizados em aplicações web.” Source: <http://dspace.unipampa.edu.br:8080/jspui/handle/rii/5512>, May, 2021.
- [43] Liu, X.; Xu, M.; Teng, T.; Huang, G.; Mei, H. “MUIT: A domain-specific language and its middleware for adaptive mobile web-based user interfaces in ws-bpel”, *IEEE Transactions on Services Computing*, vol. 12–6, Dec 2019, pp. 955–969.
- [44] Logre, I.; Déry-Pinna, A.-M. “Mde in support of visualization systems design: A multi-staged approach tailored for multiple roles”, *Human-Computer Interaction*, vol. 2, Jun 2018, pp. 17.
- [45] Luger, E.; Sellen, A. “Like having a really bad pa: The gulf between user expectation and experience of conversational agents”. In: Conference on Human Factors in Computing Systems, 2016, pp. 5286–5297.
- [46] Marchezan, L.; Bolfe, G.; Rodrigues, E.; Bernardino, M.; Basso, F. P. “Thoth: A web-based tool to support systematic reviews”. In: International Symposium on Empirical Software Engineering and Measurement, 2019, pp. 1–6.
- [47] McCabe, T. J. “A complexity measure”, *IEEE Transactions on Software Engineering*, vol. SE-2–4, Dec 1976, pp. 308–320.
- [48] Mernik, M.; Heering, J.; Sloane, A. M. “When and how to develop domain-specific languages”, *ACM Computing Surveys*, vol. 37–4, Dec 2005, pp. 316–344.

- [49] Ministério da Saúde - Conselho Nacional de Saúde do Brasil. “Resolução n. 466, de 12 de dezembro de 2012: Aprova diretrizes e normas regulamentadoras de pesquisas envolvendo seres humanos”. Source: <https://conselho.saude.gov.br/resolucoes/2012/Reso466pdf/>, Nov, 2019.
- [50] Mosqueira-Rey, E.; Alonso-Ríos, D. “Usability heuristics for domain-specific languages (DSLs)”. In: 35th Annual ACM Symposium on Applied Computing, 2020, pp. 1340—1343.
- [51] Msosa, Y. J. “FCIG grammar evaluation: A usability assessment of clinical guideline modelling constructs”. In: IEEE Symposium on Computers and Communications, 2019, pp. 1141–1146.
- [52] Nandra, C.; Gorgan, D. “Usability evaluation of a domain specific language for defining aggregated processing tasks”. In: 15th International Conference on Intelligent Computer Communication and Processing, 2019, pp. 87–94.
- [53] Nielsen, J. “10 Usability Heuristics for User Interface Design”. Source: <https://www.nngroup.com/articles/ten-usability-heuristics/>, Mar, 2018.
- [54] Nielsen, J. “Usability Engineering”. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993.
- [55] Nielsen, J.; Molich, R. “Heuristic evaluation of user interfaces”. In: Conference on Human Factors in Computing Systems, 1990, pp. 249–256.
- [56] Norman, D. “The Design of Everyday Things”. New York, NY, USA: Basic Books, 2013, revised and expanded ed.
- [57] Nosal, M.; Porubán, J.; Sulír, M. “Customizing host ide for non-programming users of pure embedded DSLs: A case study”, *Computer Languages, Systems and Structures*, vol. 49, Sep 2017, pp. 101 – 118.
- [58] Object Management Group. “Software and systems process engineering metamodel specification - spem”. Source: <http://www.omg.org/spec/SPEM>, Nov, 2018.
- [59] Object Management Group. “Business Process Model and Notation - BPMN”. Source: <https://www.omg.org/spec/BPMN/2.0/PDF>, Nov, 2019.
- [60] Poltronieri, I.; Pedroso, A. C.; Zorzo, A. F.; Bernardino, M.; de Borba Campos, M. “Is usability evaluation of DSL still a trending topic?” In: *Human-Computer Interaction. Theory, Methods and Tools*, Springer International Publishing, 2021, pp. 299–317.
- [61] Poltronieri, I.; Pedroso, A. C.; Zorzo, A. F.; Bernardino, M.; de Borba Campos, M. “Is usability evaluation of DSL still a trending topic?” In: 23rd International Human-Computer Interaction, 2021, pp. 299–317.

- [62] Poltronieri, I.; Zorzo, A. F.; Bernardino, M.; Borba Campo, M. “Usa-dsl: Usability evaluation framework for domain-specific languages”. In: 33rd Symposium On Applied Computing, 2018, pp. 2013–2021.
- [63] Poltronieri, I.; Zorzo, A. F.; Bernardino, M.; de Borba Campos, M. “Usability evaluation framework for domain-specific language: A focus group study”, *Applied Computing Review*, vol. 18–3, Oct 2018, pp. 5–18.
- [64] Potronieri, I.; Zorzo, A.; Bernardino.M.; Medeiros.B.; Campos, M. “Heuristic evaluation checklist for domain-specific languages”. In: 4th International Conference on Human Computer Interaction Theory and Applications., 2020, pp. 37 – 48.
- [65] Prates, R. O.; Barbosa, S. D. J. “Avaliação de interfaces de usuário–conceitos e métodos”, *Jornada de Atualização em Informática do Congresso da Sociedade Brasileira de Computação, Capítulo 6*, vol. 6, Jan 2003, pp. 49.
- [66] Preece, Jennifer; Sharp, H.; Rogers, Y. “Interaction Design: Beyond Human-Computer Interaction”. New York, NY, USA: Wiley, 2015, 4th ed.
- [67] Preece, J.; Rombach, H. D. “A taxonomy for combining software engineering and human-computer interaction measurement approaches: towards a common framework”, *International journal of human-computer studies*, vol. 41–4, Oct 1994, pp. 553–583.
- [68] Pressman, R. “Software Engineering: A Practitioner’s Approach”. New York, NY, USA: McGraw-Hill, Inc., 2010, 7th ed.
- [69] Quiñones, D.; Rusu, C.; Rusu, V. “A methodology to develop usability/user experience heuristics”, *Computer Standards and Interfaces*, vol. 59, Aug 2018, pp. 109 – 129.
- [70] Rodrigues, I.; Campos, M. B.; Zorzo, A. F. “Usability Evaluation of Domain-Specific Languages: a Systematic Literature Review”. In: 19th International Conference on Human-Computer Interaction, 2017, pp. 522–534.
- [71] Rodrigues, I. P.; Zorzo, A. F.; Bernardino, M. “Uma proposta de processo de avaliação de usabilidade para dsls”. In: Anais da III Escola Regional de Engenharia de Software, 2019, pp. 127–136.
- [72] Rodriguez-Gil, L.; Garcia-Zubia, J.; Orduña, P.; Villar-Martinez, A.; López-De-Ipina, D. “New approach for conversational agent definition by non-programmers: A visual domain-specific language”, *IEEE Access*, vol. 7, Apr 2019, pp. 5262–5276.
- [73] Rogers, Y.; Sharp, H.; Preece, J. “Design de interação: além da interação humano-computador”. Porto Alegre, RS, Brasil: Bookman, 2013, 3th ed.
- [74] Rosenzweig, E. “Successful User Experience: Strategies and Roadmaps”. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2015.

- [75] Rouly, J. M.; Orbeck, J. D.; Syriani, E. “Usability and suitability survey of features in visual IDEs for non-programmers”. In: 5th Workshop on Evaluation and Usability of Programming Languages and Tools, 2014, pp. 31–42.
- [76] Sharp, H.; Preece, J.; Rogers, Y. “Interaction Design: Beyond Human - Computer Interaction”. New York, NY, USA: Wiley Publishing, 2019, 5th ed.
- [77] Shin, S.-S. “Empirical study on the effectiveness and efficiency of model-driven architecture techniques”, *Software and Systems Modeling*, vol. 18–5, Jan 2019, pp. 3083–3096.
- [78] Silva, J.; Barisic, A.; Amaral, V.; Goulão, M.; Tezel, B.; Alaca, O.; Challenger, M.; Kardas, G. “Comparing the usability of two multi-agents systems DSLs: Sea_ml++ and DSML4MAS study design”. In: 3rd International Workshop on Human Factors in Modeling, 2018, pp. 770–777.
- [79] Sinha, A.; Smidts, C. “An experimental evaluation of a higher-ordered-typed-functional specification-based test-generation technique”, *Empirical Software Engineering*, vol. 11–2, Jun 2006, pp. 173–202.
- [80] Sommerville, I. “Software Engineering”. Boston, MA, USA: Addison-Wesley, 2001, 773p.
- [81] Steenweg, R.; Kuhrmann, M.; Fernández, D. M. “Software engineering process metamodel”. In: *Software Process Modeling*, Springer, 2014, pp. 1 – 45.
- [82] Stone, D.; Jarrett, C.; Woodroffe, M.; Minocha, S. “User Interface Design and Evaluation”. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2005.
- [83] Teruel, M. A.; Navarro, E.; López-Jaquero, V.; Montero, F.; González, P. “A cscw requirements engineering case tool: Development and usability evaluation”, *Information and Software Technology*, vol. 56–8, Aug 2014, pp. 922 – 949.
- [84] U.S. General Services Administration/Technology Transformation Services. “Usability Gov Improving the User Experience”. Source: <https://www.usability.gov/>, Apr, 2017.
- [85] Van Deursen, A.; Klint, P.; Visser, J. “Domain-specific languages: An annotated bibliography”, *ACM SIGPLAN Notes*, vol. 35–6, Jun 2000, pp. 26–36.
- [86] Wohlin, C.; Runeson, P.; Host, M.; Ohlsson, M. C.; Regnell, B.; Wesslen, A. “Experimentation in Software Engineering: An Introduction”. Norwell, MA, USA: Springer - Verlag Berlin Heidelberg, 2014.

APPENDIX A – STEPS - USA-DSL PROCESS

A Step is composed of concepts, which are similar concerns and work efforts, that is, it is a collection of Tasks related to certain Activities and executed for a set of profiles called Role Sets. In this appendix we present the list of Steps that compose our process. The complete description can be found in Section 2.5.

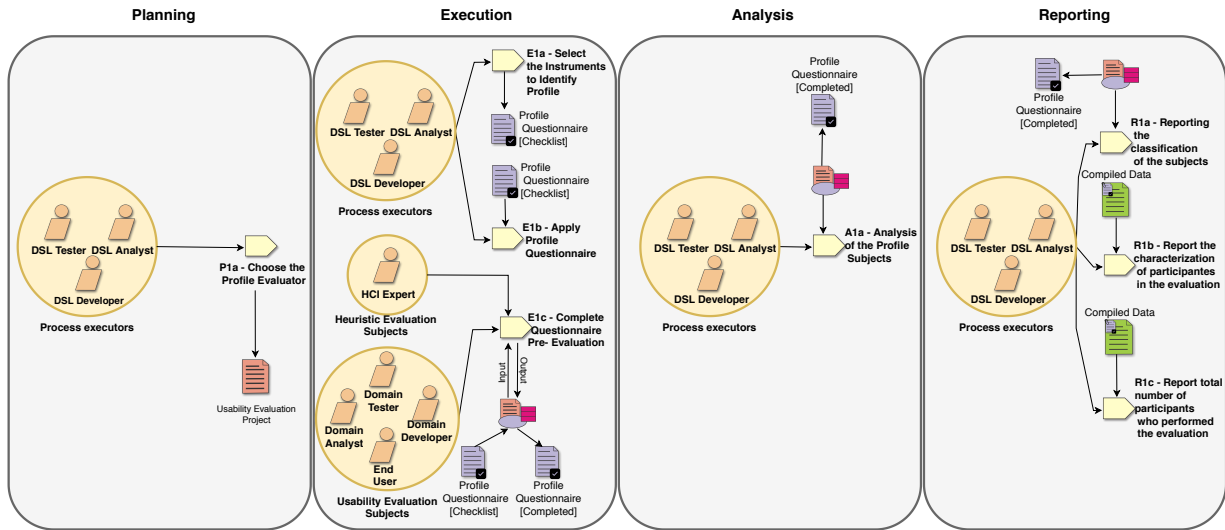


Figure A.1: Step 1 - Evaluators Profiles

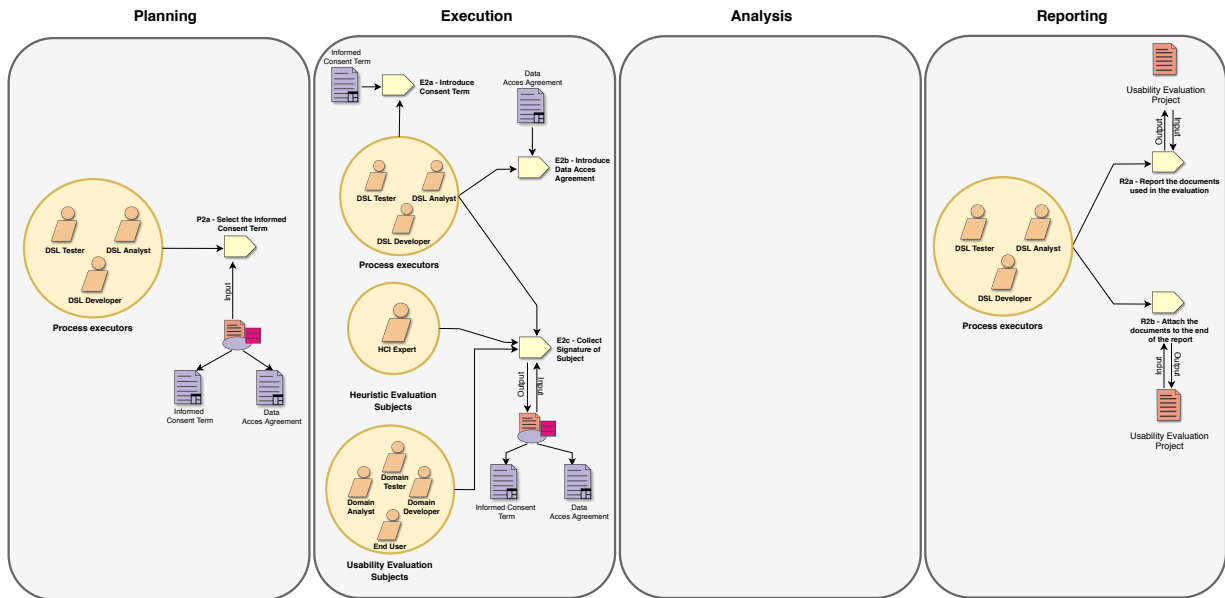


Figure A.2: Step 2 - Ethical and Legal Responsibilities

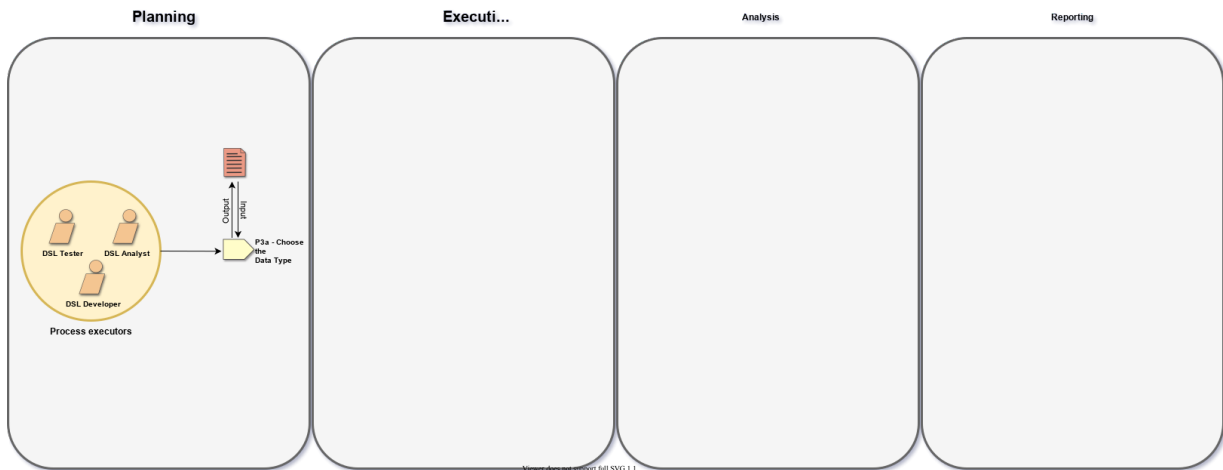


Figure A.3: Step 3 - Data Type

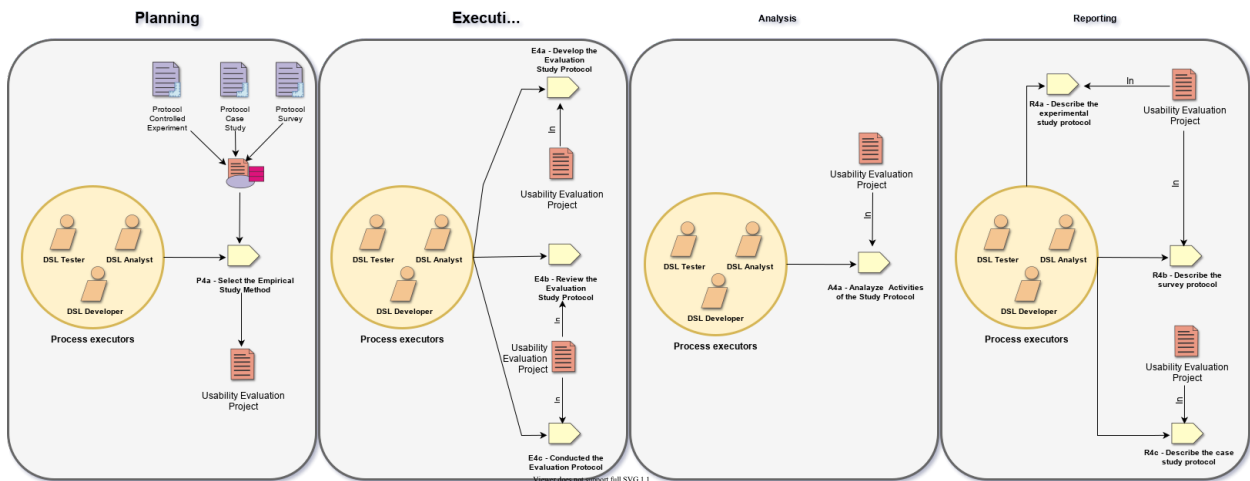


Figure A.4: Step 4 - Empirical Study Method (SE)

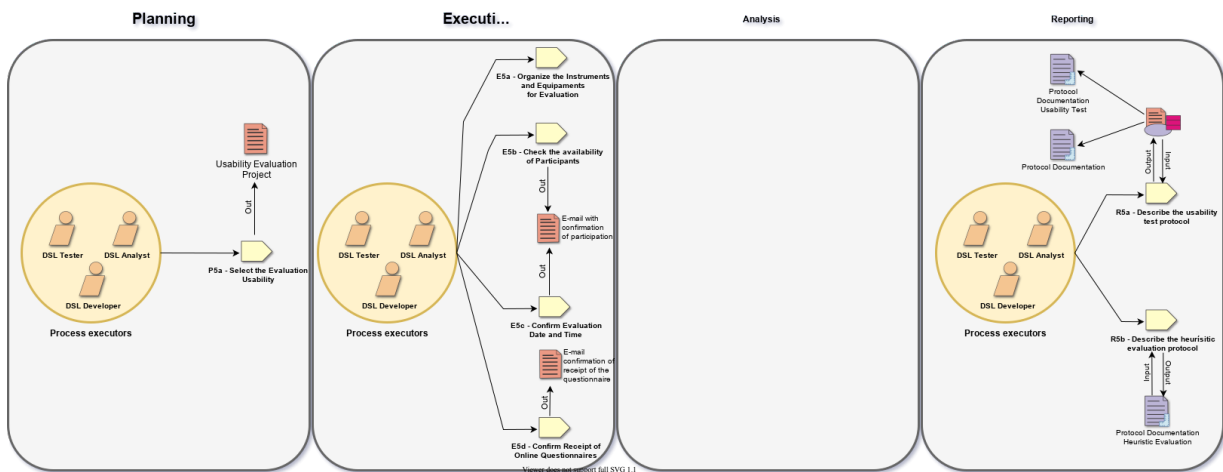


Figure A.5: Step 5 - Evaluation Method (HCI)

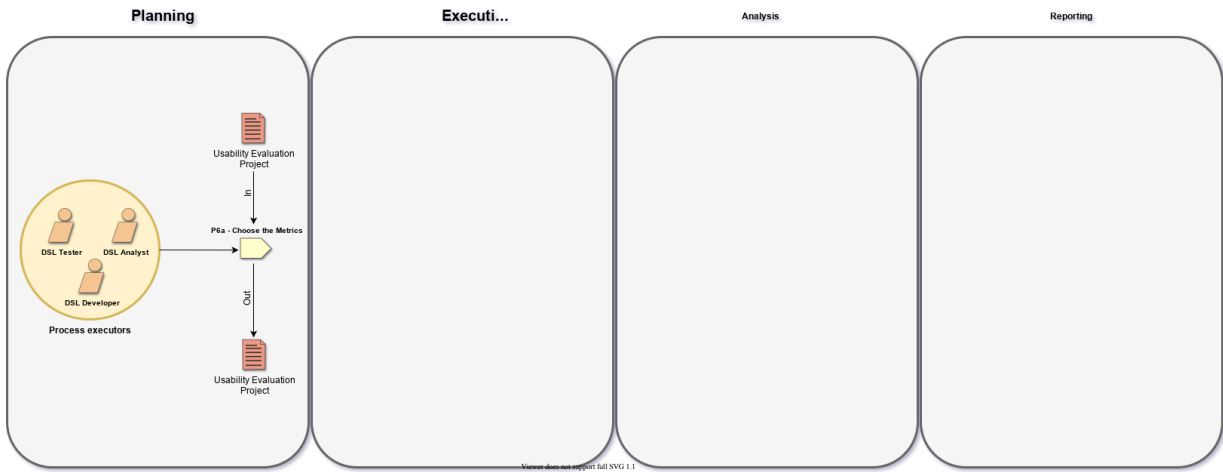


Figure A.6: Step 6 - Metrics

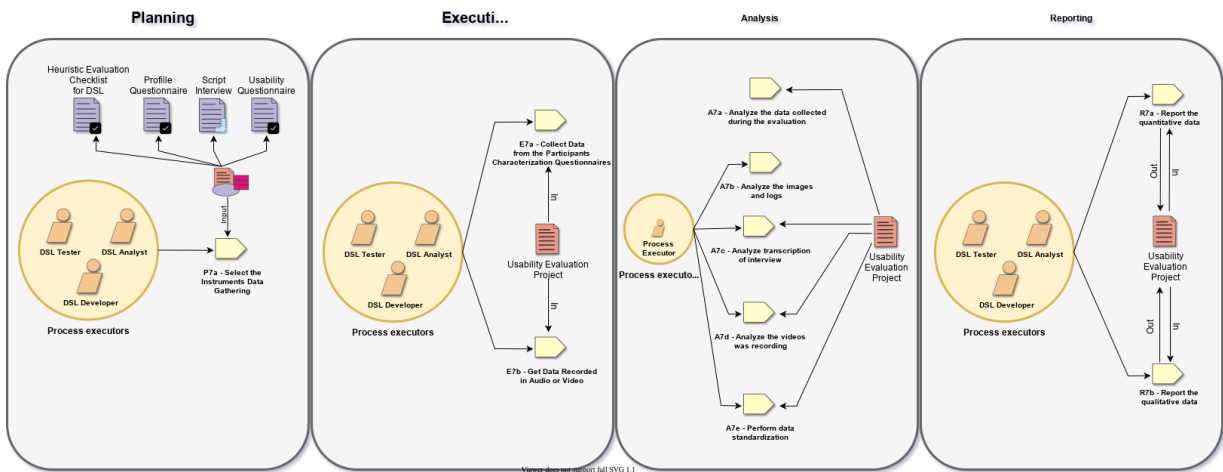


Figure A.7: Step 7 - Gathering Instruments

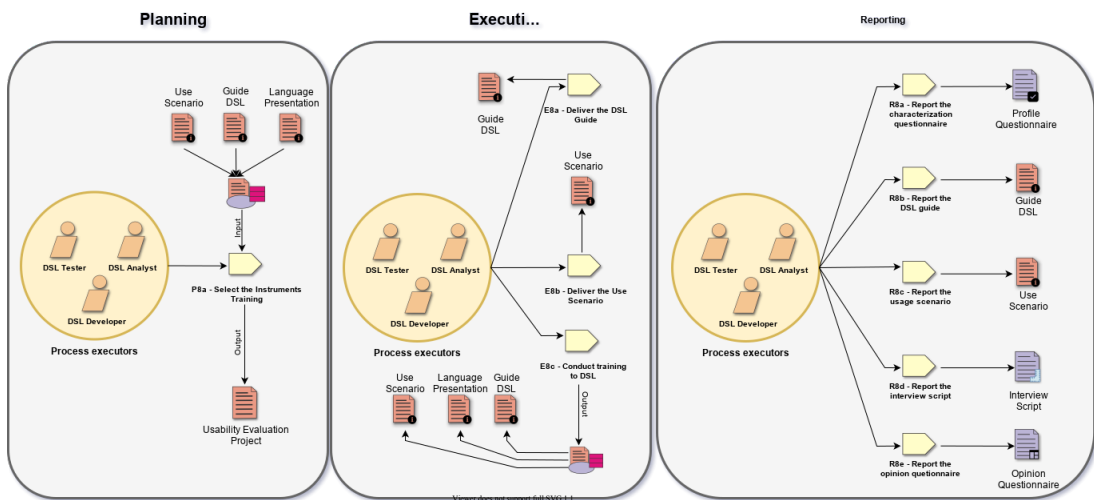


Figure A.8: Step 8 - Evaluation Instructions

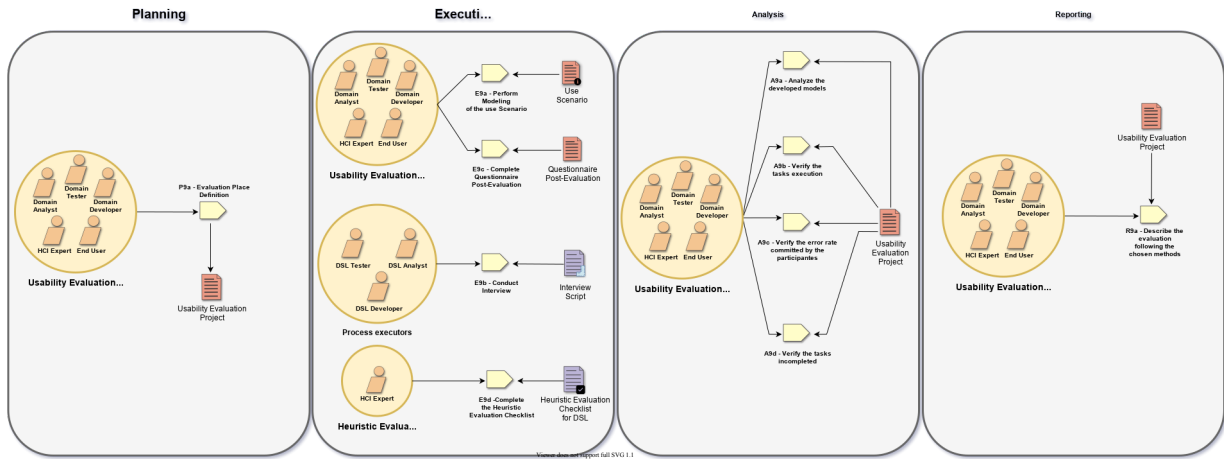


Figure A.9: Step 9 - Evaluation Conduction

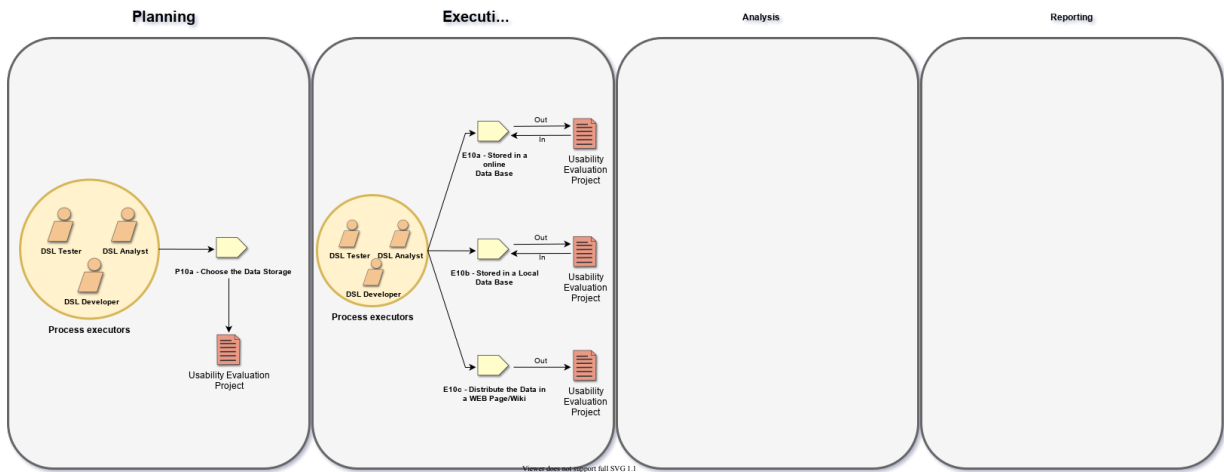


Figure A.10: Step 10 - Data Packaging

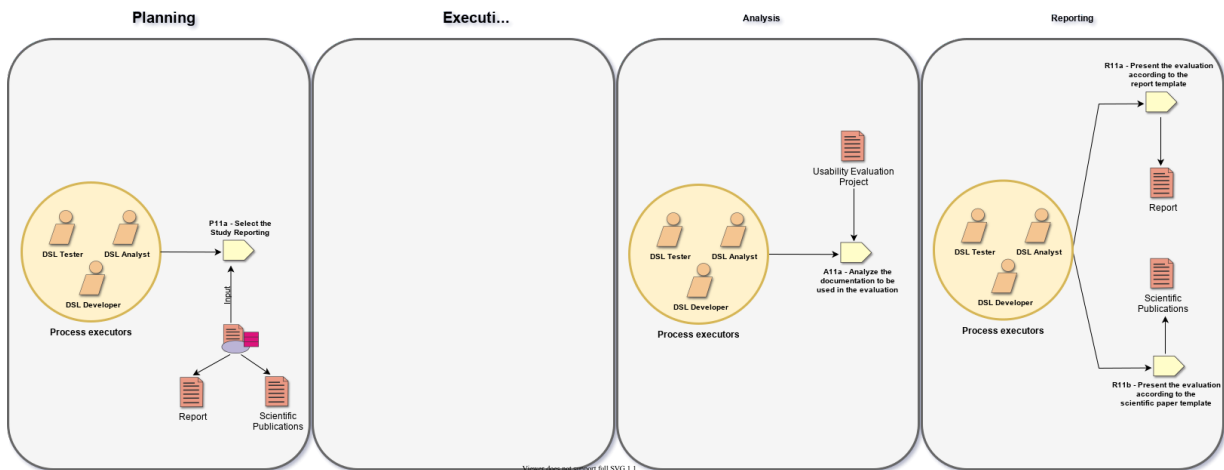


Figure A.11: Step 11 - Evaluation Reporting

APPENDIX B – ACTIVITIES - USA-DSL PROCESS

In the Usa-DSL Process, Activities group together Tasks from a Step in a Phase from the process life cycle. Hence, each Activity contains an identification composed of the initial letter of the Phase, a number that identifies a Step, and a description for the Activity.

- *Planning Phase*

P1 - Define Evaluators Profile

P2 - Define Informed Consent Term

P3 - Define Data Type

P4 - Define Empirical Study Method

P5 - Define Evaluation Usability Type

P6 - Define Metrics for Language Evaluation

P7 - Define the Instruments of Data Gathering

P8 - Define the Instruments of Instructions and Training

P9 - Define Execution Place

P10 - Define Data Storage

P11 - Define Study Reporting

- *Execution Phase*

E1 - Apply Instruments to identify Profiles

E2 - Introduce the Form and Collects Signatures of Subject

E4 - Develop and Conduct Protocol

E5- Prepare the Evaluation

E7 - Data Collection

E8 - Introduce Instruments of Instruction and Conduct Training

E9 - Execution of s and Evaluation Conduction

E10 - Store Data Obtained

- *Analysis Phase*

A1 - Analyze Evaluator Profiles

A4 - Analyze the Developed Protocol

A7 - Analyze the Collected Data

A9 - Analyze the Performed Tasks

A11 - Analyze the Documentation

- *Reporting Planning*

R1 - Report Evaluator Profiles

R2 - Report Subjects Number and the Form Used

R4 - Report the Develop Protocol

R5 - Report Conduction Evaluation

R7 - Report Data Analysis

R8 - Report the Instruments

R9 - Report Tasks Analysis

R11 - Report the Results and Analyzed Information

APPENDIX C – TASKS - USA-DSL PROCESS

The Tasks that compose of Usa-DSL Process will be presented in the context of an Activity as defined in Section 4.1:

- *P1 - Define Evaluators Profile* - this Activity contains one Task (Figure C.1):

P1a - Choose the Profile Evaluator

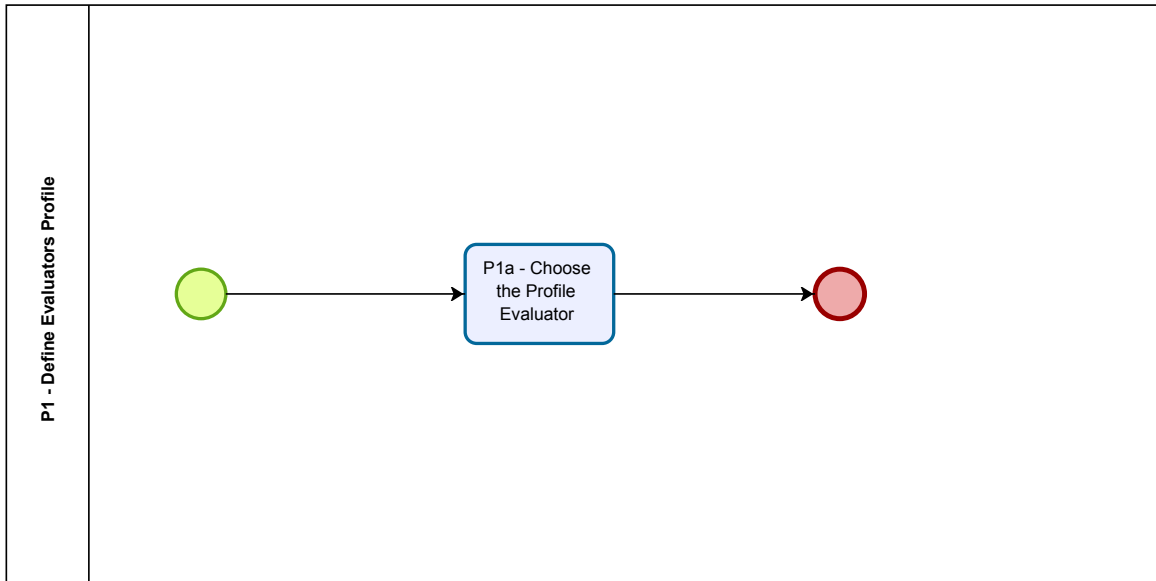


Figure C.1: Task P1a - Choose the Profile Evaluator

- *E1 - Apply Instruments to Identify Profiles*: this Activity contains three Tasks (Figure C.2):

E1a - Select the Instruments to Identify Profile

E1b - Apply Profile Questionnaire

E1c - Complete Questionnaire Pre-Evaluation

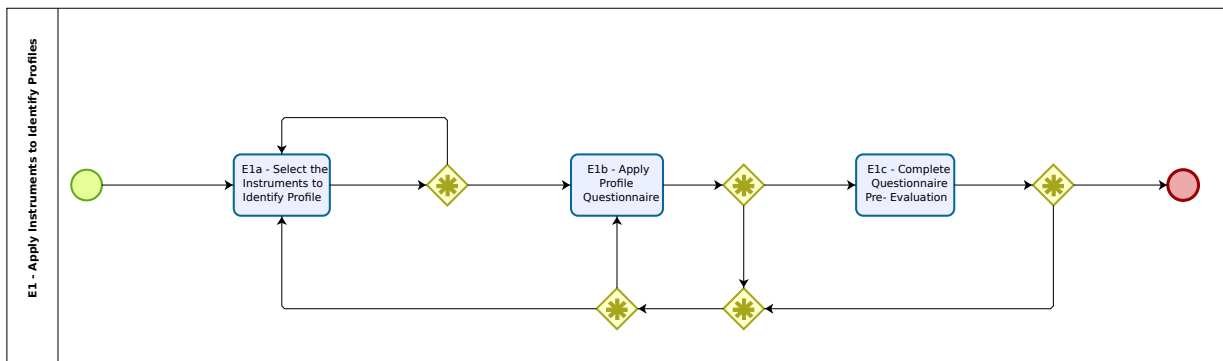


Figure C.2: Tasks of E1 - Apply Instruments to Identify Profiles

- *A1 - Analyze Evaluator Profiles*: this Activity contains one Task (Figure C.3):

A1a - Analysis of Profile Subjects

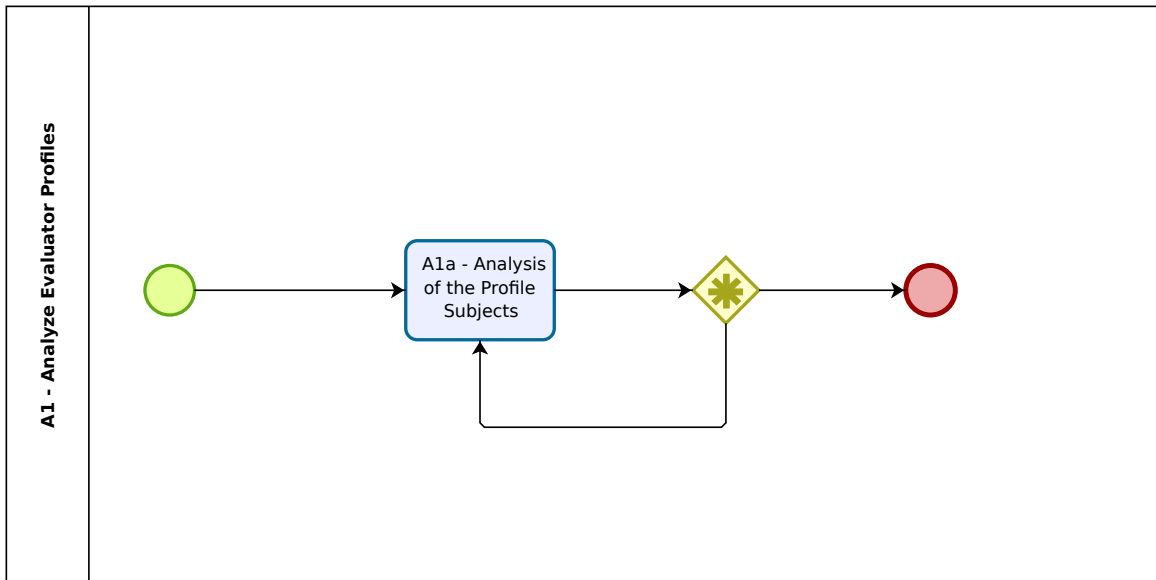


Figure C.3: Task A1a - Analysis of Profile Subjects

- *R1 - Report Evaluator Profiles*: this Activity contains three Tasks (Figure C.4):

R1a - Reporting the Classification of the Subjects

R1b - Reporting the Characterization of Participants

R1c - Reporting Total Number of Participants who Performed the Evaluation

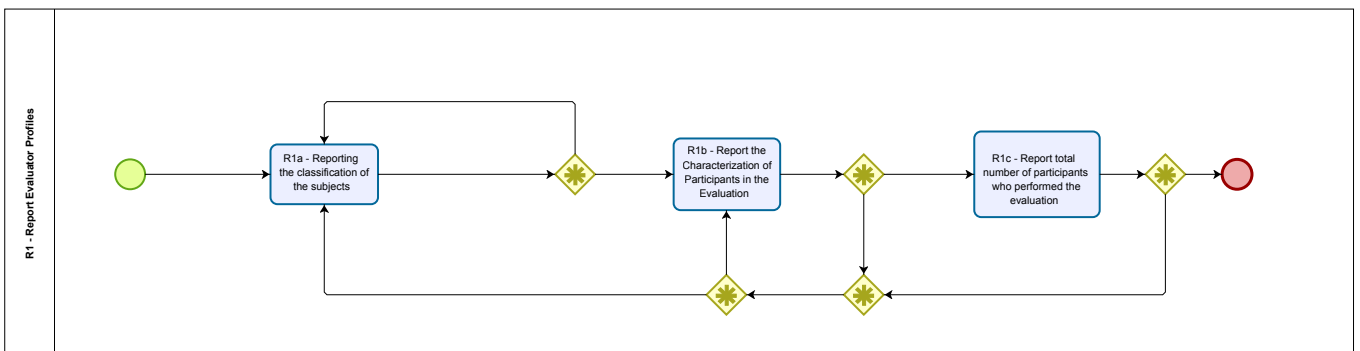


Figure C.4: Tasks of R1 - Report Evaluator Profiles

- *P2 - Define Informed Consent Term*: this Activity contains one Task (Figure C.5):

P2a - Select the Informed Consent Term

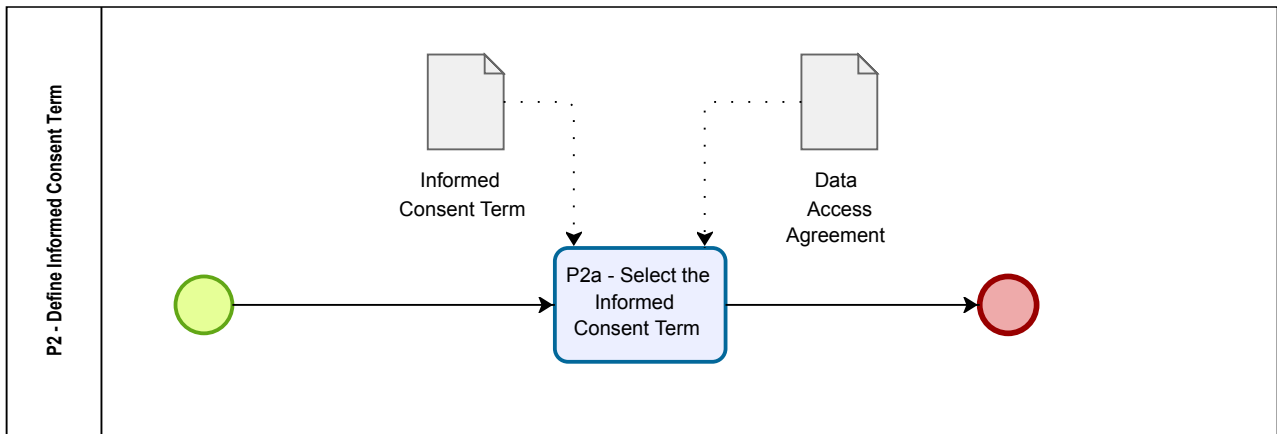


Figure C.5: Task P2a - Select the Informed Consent Term

- *E2 - Introduce the Form and Collects Signatures of Subject*: this Activity contains three Tasks (Figure C.6):

E2a - Introduce ICT

E2b - Introduce TCUD

E2c - Collected Signature of Subject

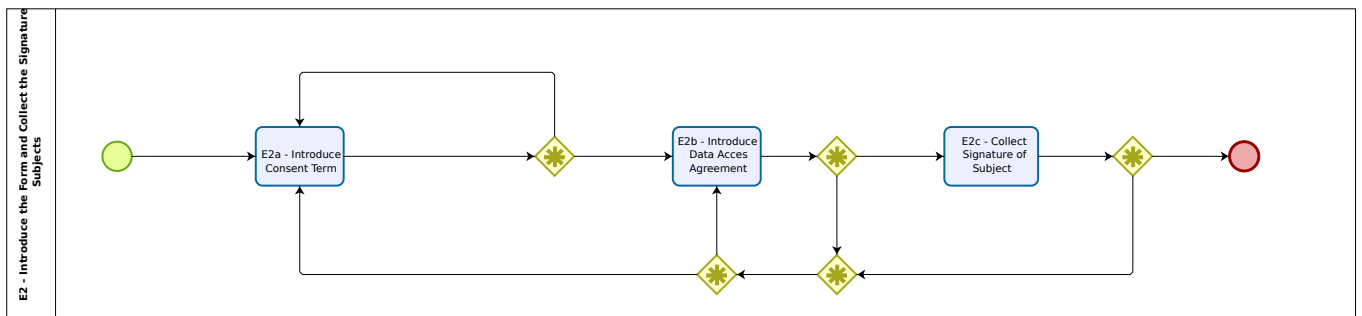


Figure C.6: Tasks of E2 - Introduce the Form and Collects Signatures of Subject

- *R2 - Report Subjects Number and the Used Form*: this Activity contains two Tasks (Figure C.7):

R2a - Reporting the Documents used in the Evaluation

R2b - Attach the Documents to the End of the Report

- *P3 - Define Data Type*: this Activity contains one Task (Figure C.8):

P3a - Choose the Data Type

- *P4 - Define Empirical Study Method*: this Activity contains one Task (Figure C.9):

P4a - Select the Empirical Study Method

- *E4 - Develop and Conduct Protocol*: this Activity contains three Tasks (Figure C.10):

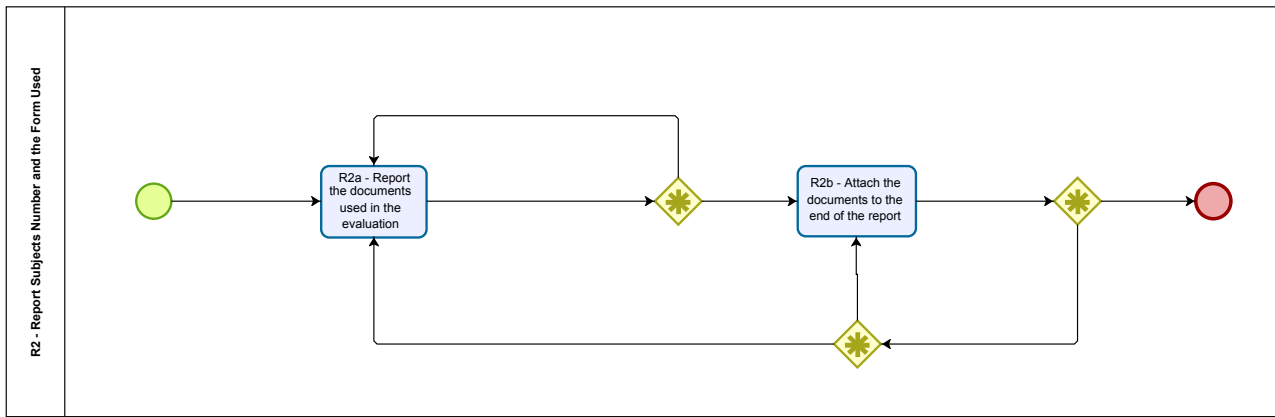


Figure C.7: Tasks of R2 - Report Subjects Number and Used Form

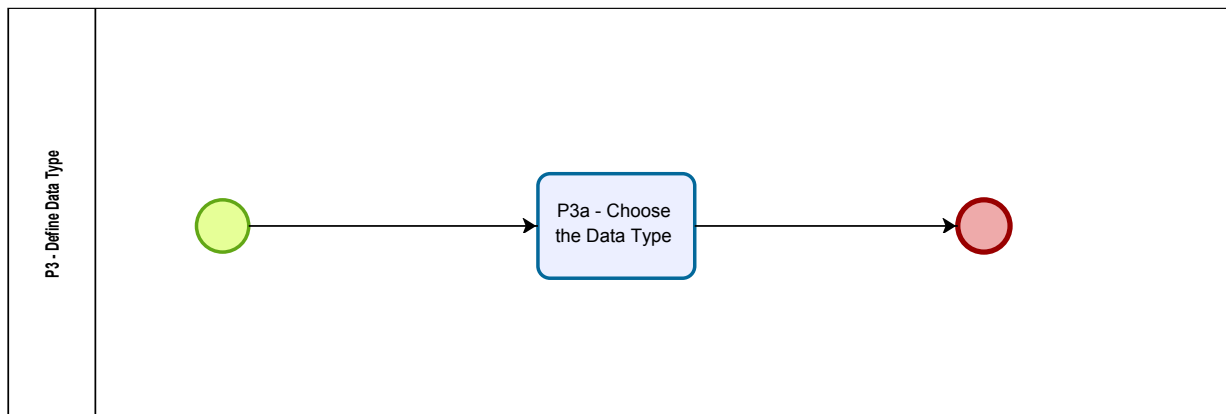


Figure C.8: Task P3a - Choose the Data Type

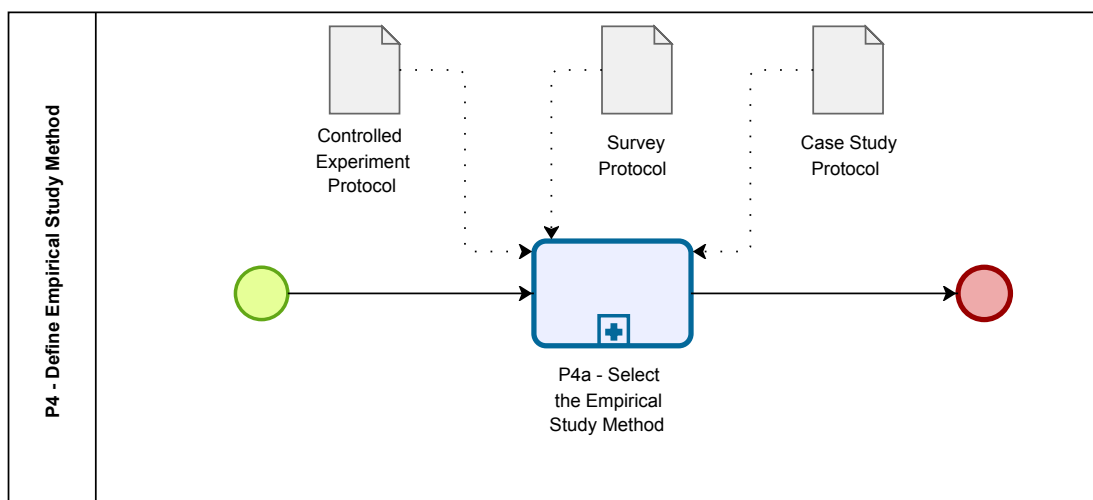


Figure C.9: Task P4a - Select the Empirical Study Method

E4a - Develop the Evaluation Study Protocol

E4b - Review the Evaluation Study Protocol

E4c - Conducted the Evaluation Protocol

- *A4 - Analyze the Developed Protocol*: this Activity contains one Task (Figure C.11):

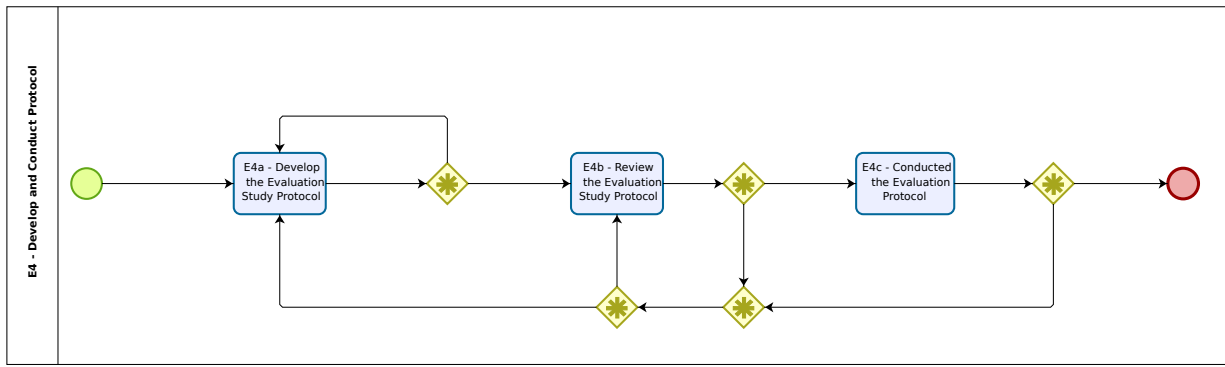


Figure C.10: Tasks of E4 - Develop and Conduct Protocol

A4a - Analyze Activities of the Study Protocol

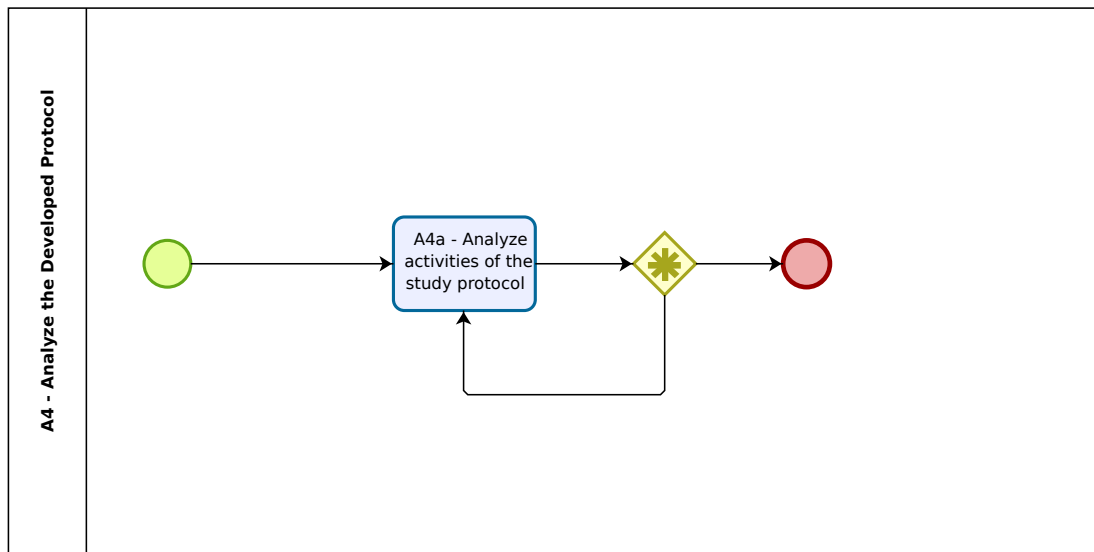


Figure C.11: Task A4a - Analyze Activities of the Study Protocol

- *R4 - Report the Developed Protocol*: this Activity contains three Tasks (Figure C.12):

R4a - Describe the Experimental Study Protocol

R4b - Describe the Survey Protocol

R4c - Describe the Case Study Protocol

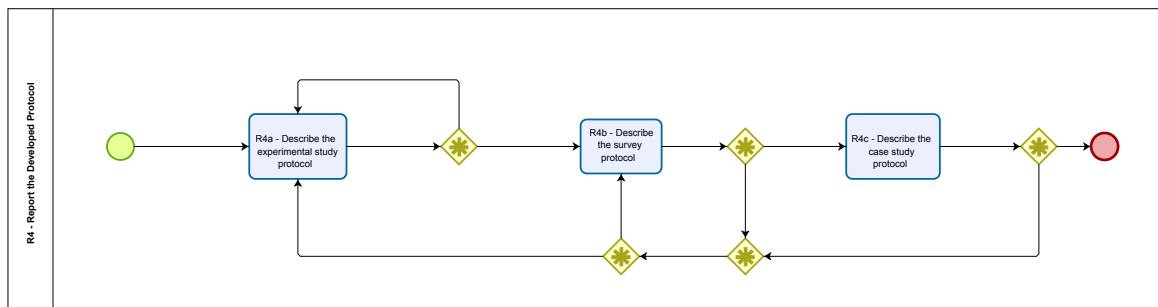


Figure C.12: Tasks of R4 - Report the Developed Protocol

- *P5 - Define Evaluation Usability Type*: This Activity contains one Task (Figure C.13):

P5a - Select the Evaluation Usability

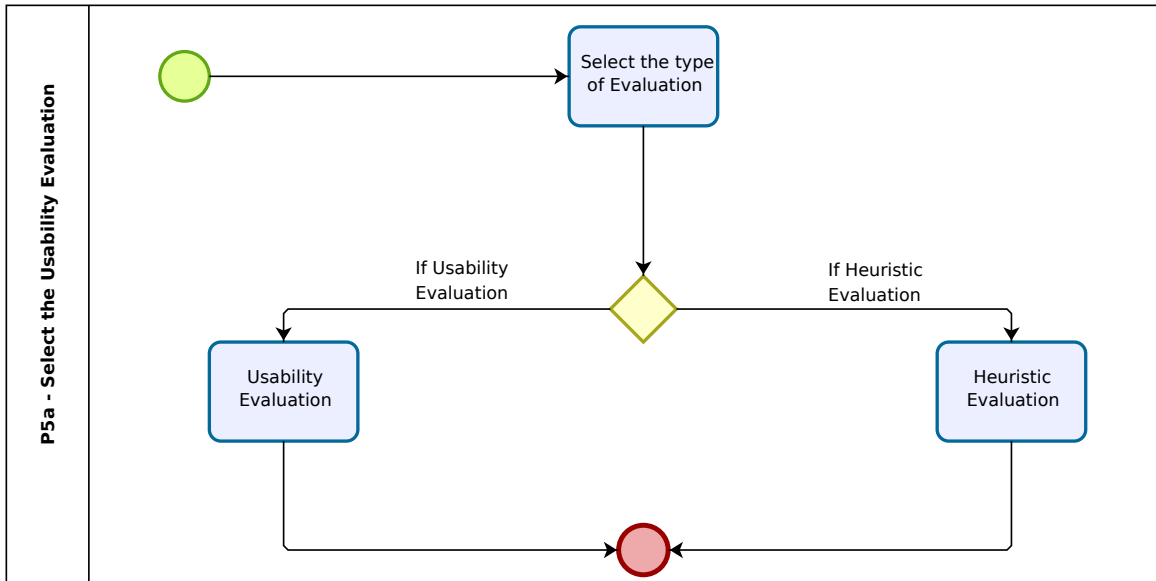


Figure C.13: Task P5a - Select the Evaluation Usability

- *E5 - Prepare the Evaluation*: this Activity contains four Tasks (Figure C.14):

E5a - Organize the Instruments and Equipments for Evaluation

E5b - Check the Availability of Participants

E5c - Confirm Evaluation Data and Time

E5d - Confirm Receipt of Online Questionnaires

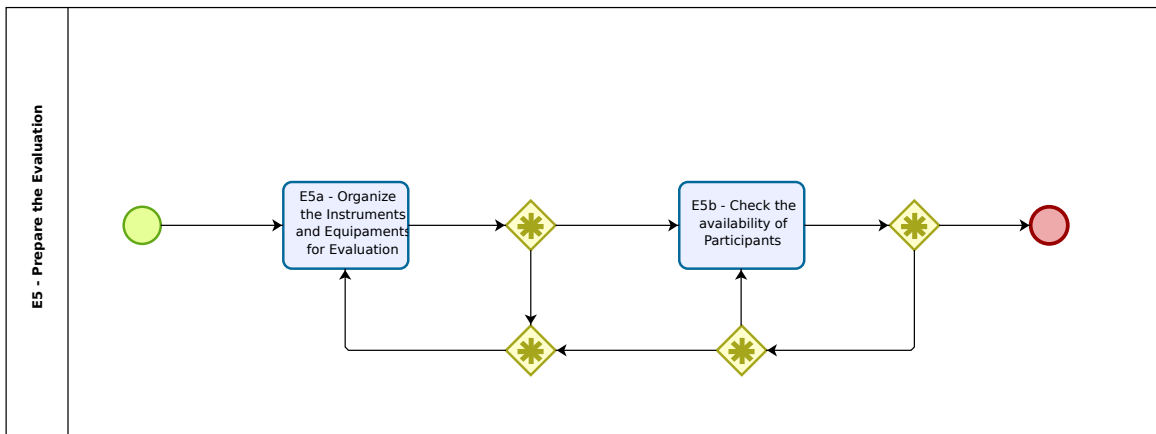


Figure C.14: Tasks of E5 - Prepare the Evaluation

- *R5 - Report Conduction Evaluation*: this Activity contains two Tasks (Figure C.15):

R5a - Describe the Usability Test Protocol

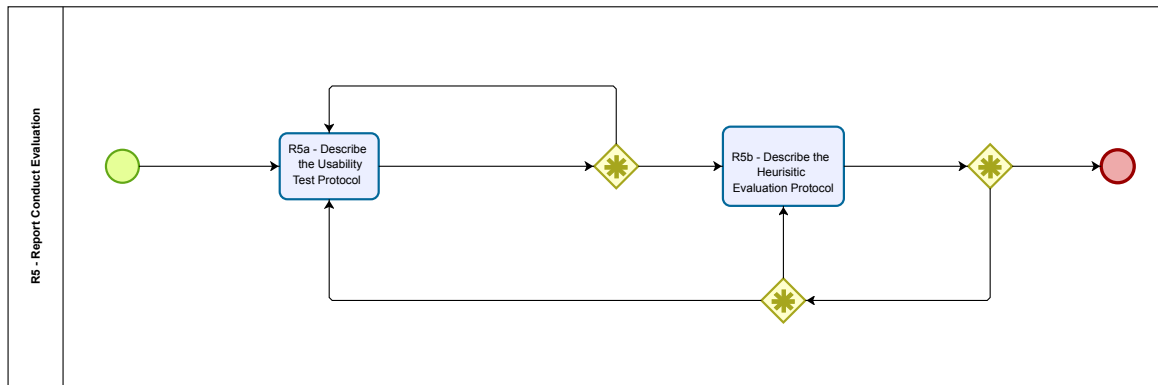


Figure C.15: Tasks of R5 - Report Conduction Evaluation

R5b - Describe the Heuristic Evaluation Protocol

- *P6 - Define Metrics for Language Evaluation*: this Activity contains one Task (Figure C.16):

P6a - Choose the Metrics

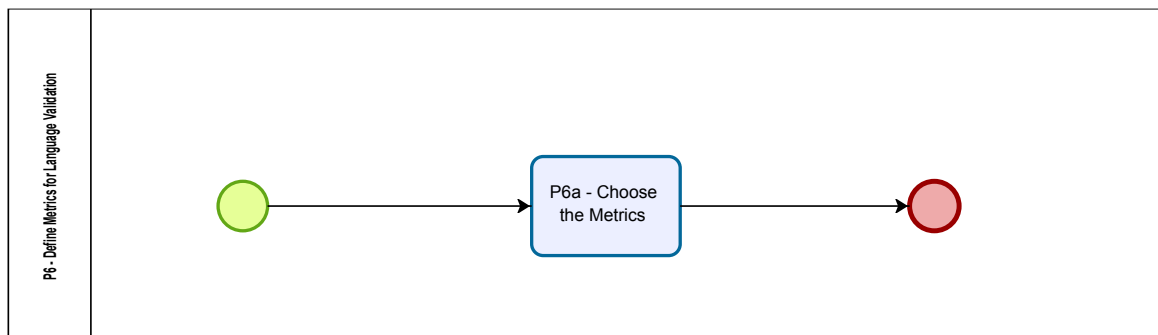


Figure C.16: Task P6a - Choose the Metrics

- *P7 - Define Instruments of Data Gathering*: this Activity has one Task (Figure C.17):

P7a - Select the Instruments Data Gathering

- *E7 - Data Collection*: this Activity contains two Tasks (Figure C.18):

E7a - Collect Data from the Participants Characterization Questionnaires

E7b - Get Data Recorded in Audio or Video

- *A7 - Analyze the Collected Data*: this Activity contains five Tasks (Figure C.19):

A7a - Analyze the Data Collected During the Evaluation

A7b - Analyze the Images and Logs

A7c - Analyze Transcription of Interview

A7d - Analyze the Videos was Recording

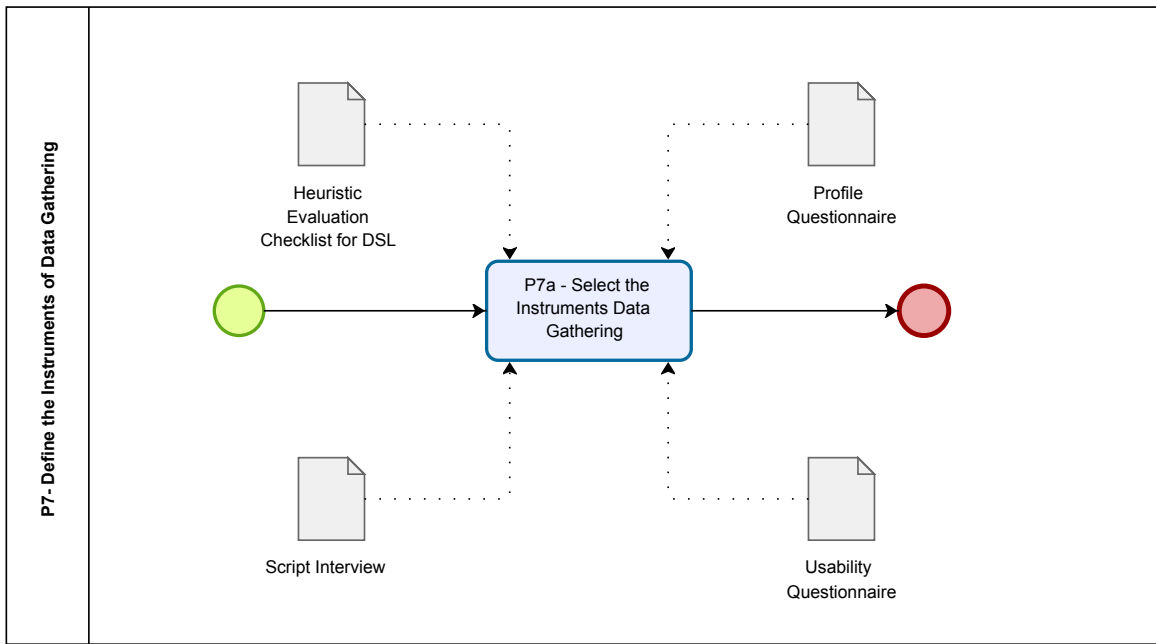


Figure C.17: Task P7a - Select the Instruments Data Gathering

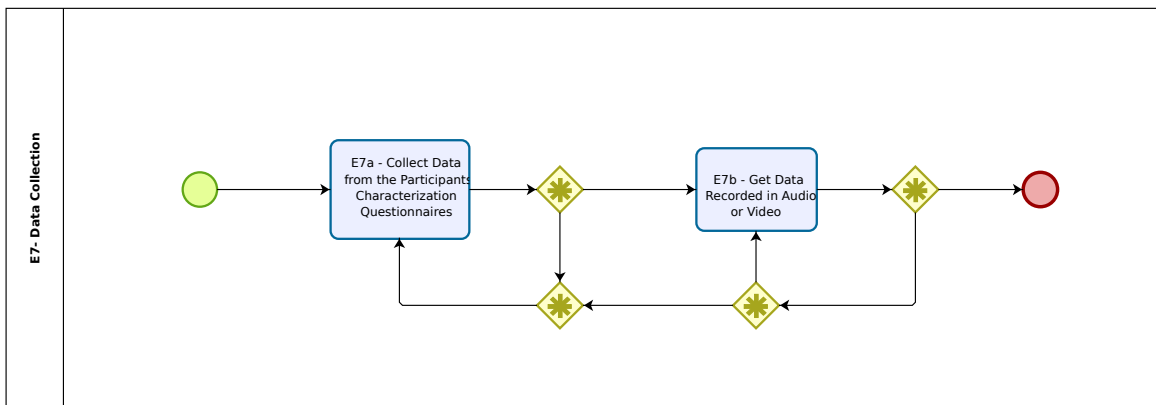


Figure C.18: Tasks of E7 - Data Collection

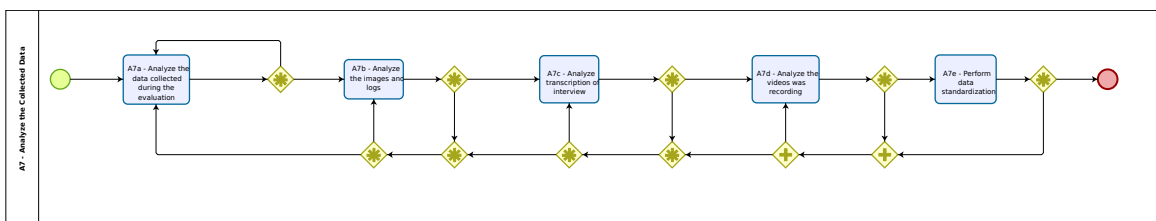


Figure C.19: Tasks of A7 - Analyze the Collected Data

A7e - Perform Data Standardization

- *R7 - Report Analysis:* this Activity contains two Tasks (Figure C.20):

R7a - Report the Quantitative Data

R7b - Report the Quantitative Data

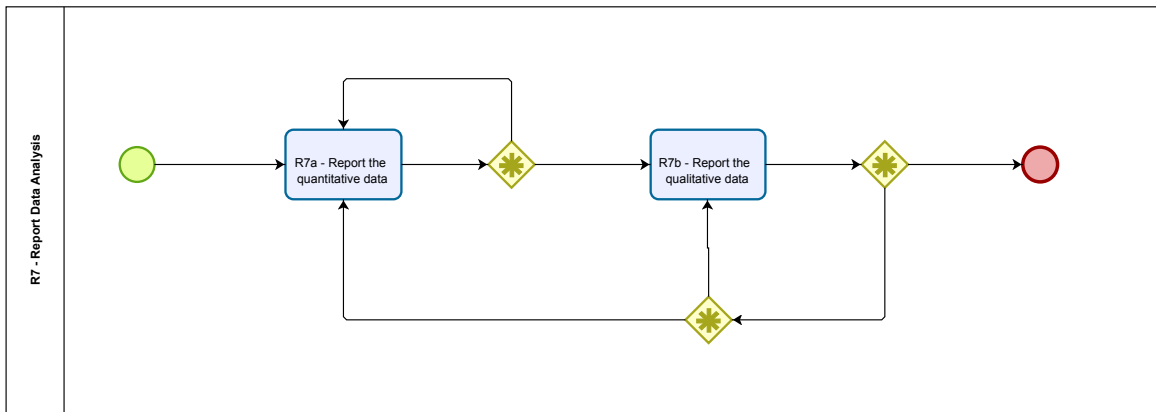


Figure C.20: Tasks of R7 - Report Analysis

- *P8 - Define the Instruments of Instructions and Training*: this Activity contains one Task (Figure C.21):

P8a - Select the Instruments Training

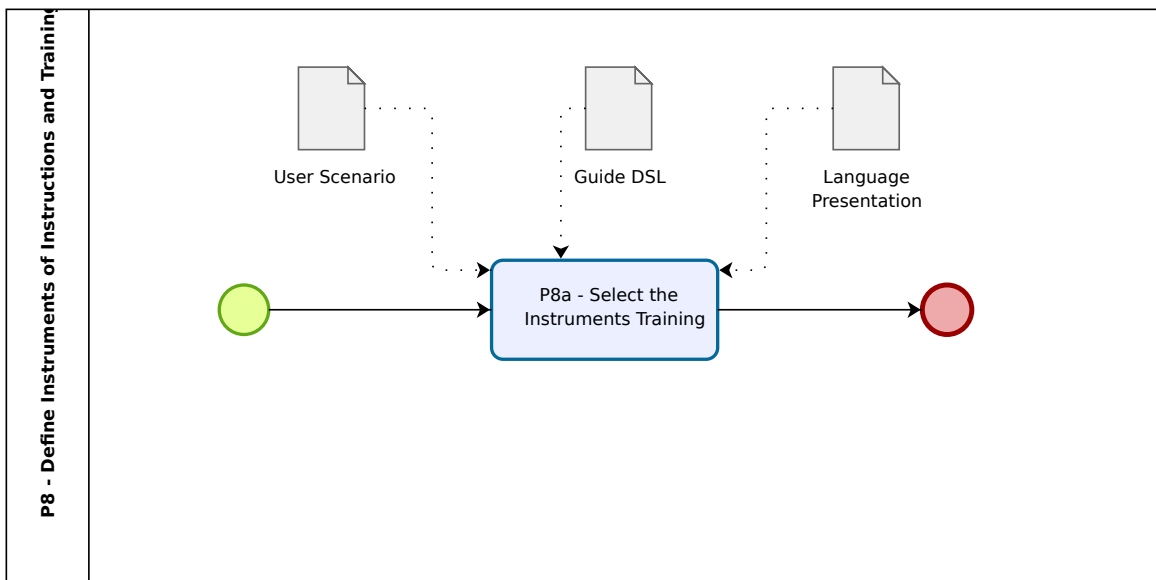


Figure C.21: Task P8a - Select the Instruments Training

- *E8 - Introduce Instruments of Instruction and Conduct Training*: this Activity contains four Tasks (Figure C.22):

E8c - Introduce the DSL

E8d - Conduct Training to DSL

E8a - Deliver the DSL Guide

E8b - Deliver the Usage Scenario

- *R8 - Report Data Analysis*: this Activity contains five Tasks (Figure C.23):

R8a - Report the Characterization Questionnaire

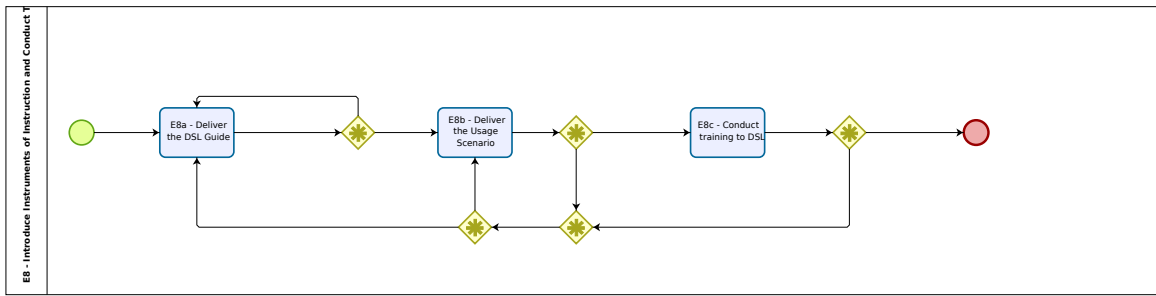


Figure C.22: Tasks of E8 - Introduce Instruments of Instruction and Conduct Training

R8b - Report the DSL Guide

R8c - Report the Usage Scenario

R8d - Report the Interview Script

R8e - Report the Opinion Questionnaire

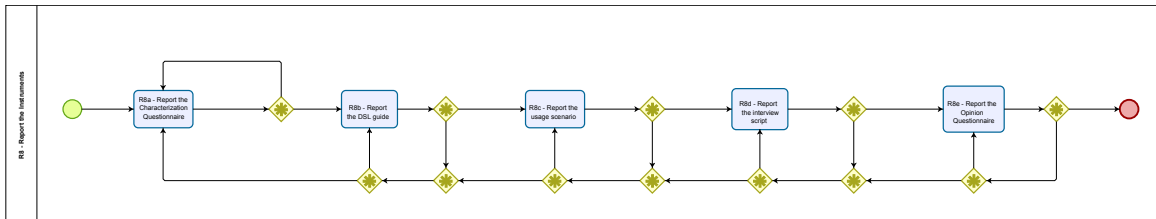


Figure C.23: Tasks of R8 - Report Data Analysis

- *P9 - Define Execution Place: this Activity contains a Tasks: (Figure C.24)*

P9a - Evaluation Place Definition

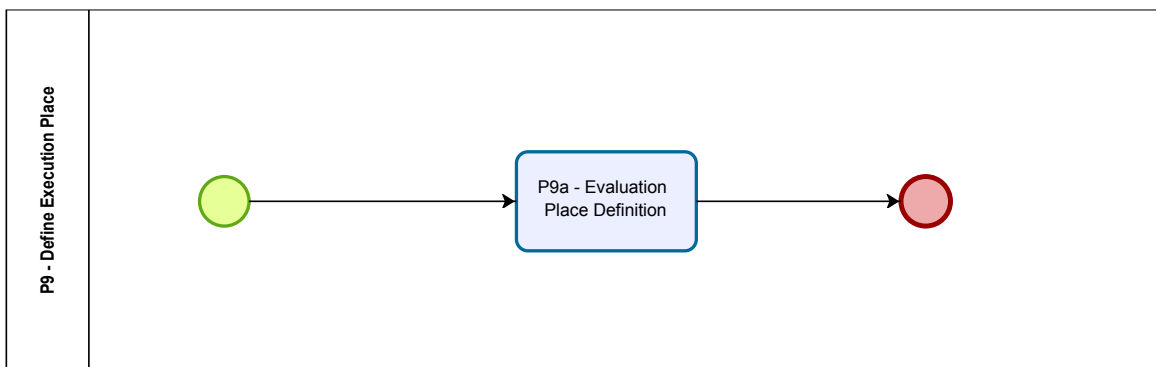


Figure C.24: Task P9a - Evaluation Place Definition

- *E9 - Execution and Evaluation Conduction: this Activity contains three Tasks (Figure C.25):*

E9a - Perform Modeling of the use Scenario

E9b - Conduct Interview

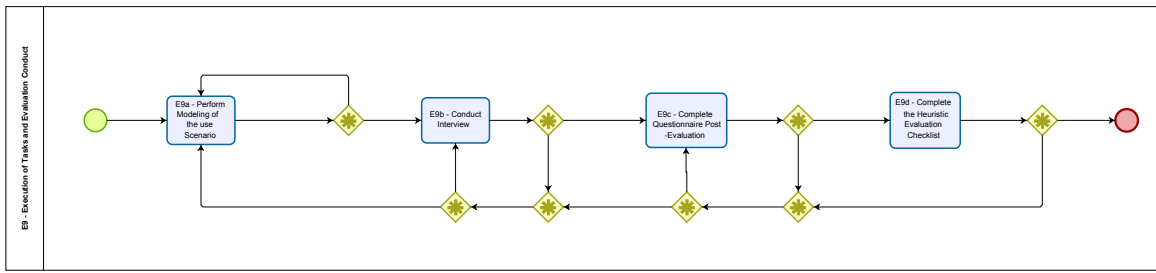


Figure C.25: Tasks of E9 - Execution and Evaluation Conduction

E9c - Complete Questionnaire Post-Evaluation

- *A9 - Analyze the Performed Tasks*: this Activity contains four Tasks (Figure C.26):

A9a - Analyze the Developed Models

A9b - Verify the Execution

A9c - Verify the Error Rate Committed by the Participants

A9d - Verify the Uncompleted Tasks

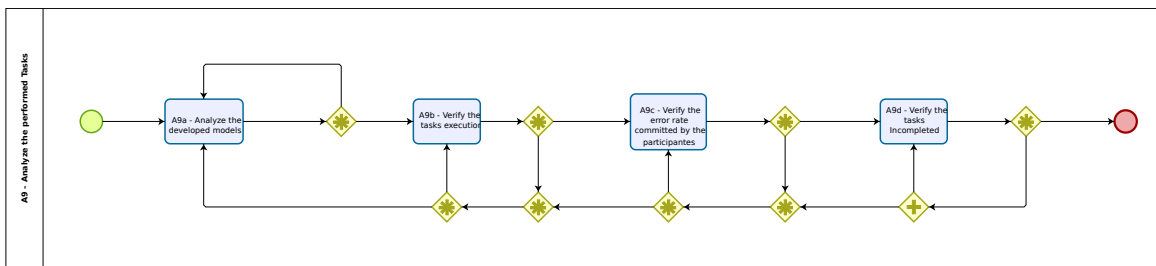


Figure C.26: Tasks of A9 - Analyze the Performed Tasks

- *R9 - Report Analysis*: this Activity contains a Task (Figure C.27):

R9a - Describe the Evaluation Following the Chosen Methods

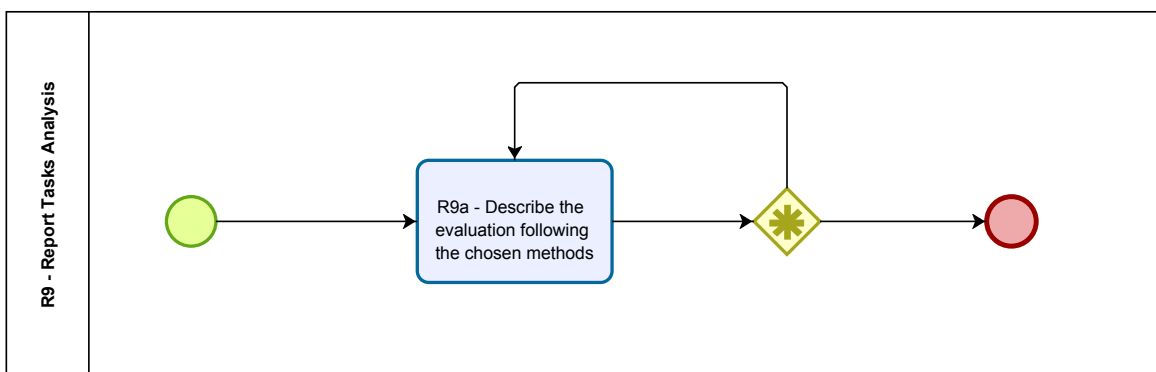


Figure C.27: Tasks R9a - Describe the Evaluation Following the Chosen Methods

- *P10 - Define Data Storage*: this Activity contains a Task (Figure C.28):

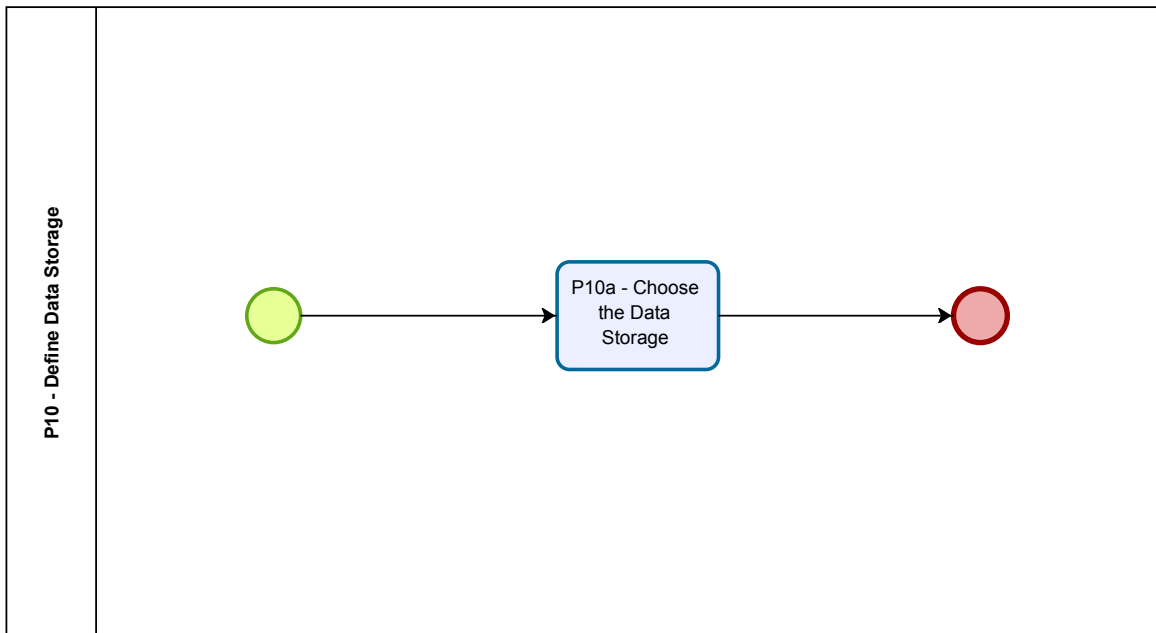


Figure C.28: Task P10a - Choose the Data Storage

P10a - Choose the Data Storage

- *E10 - Store Data Obtained*: this Activity contains three Tasks (Figure C.29):

E10a - Stored in a Online Data Base

E10b - Stored in a Local Data Base

E10c - Distribute the Data in a WEB Page/Wiki

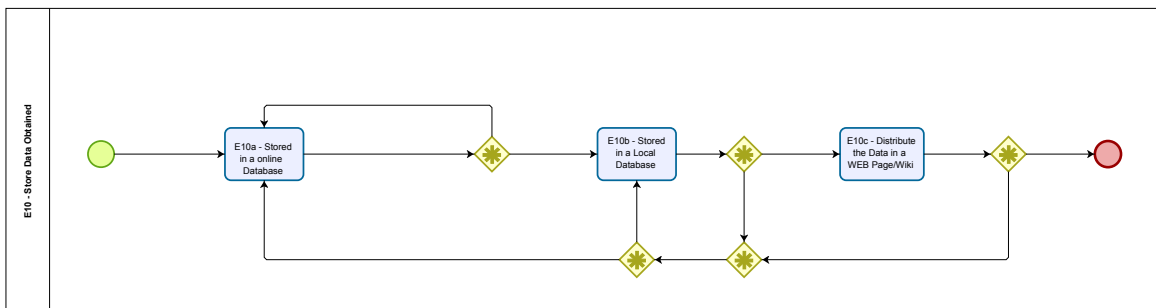


Figure C.29: Tasks of E10 - Store Data Obtained

- *P11 - Define Study Reporting*: this Activity contains a Task (Figure C.30):
 - P11a - Select the Study Reporting*
- *A11 - Analyze the Documentation* This Activity contains a Task (Figure C.31):
 - A11a - Analyze the Documentation to be Used in the Evaluation*
- *R11 - Report the Results and Analyzed Information*: this Activity contains two Tasks (Figure C.32):

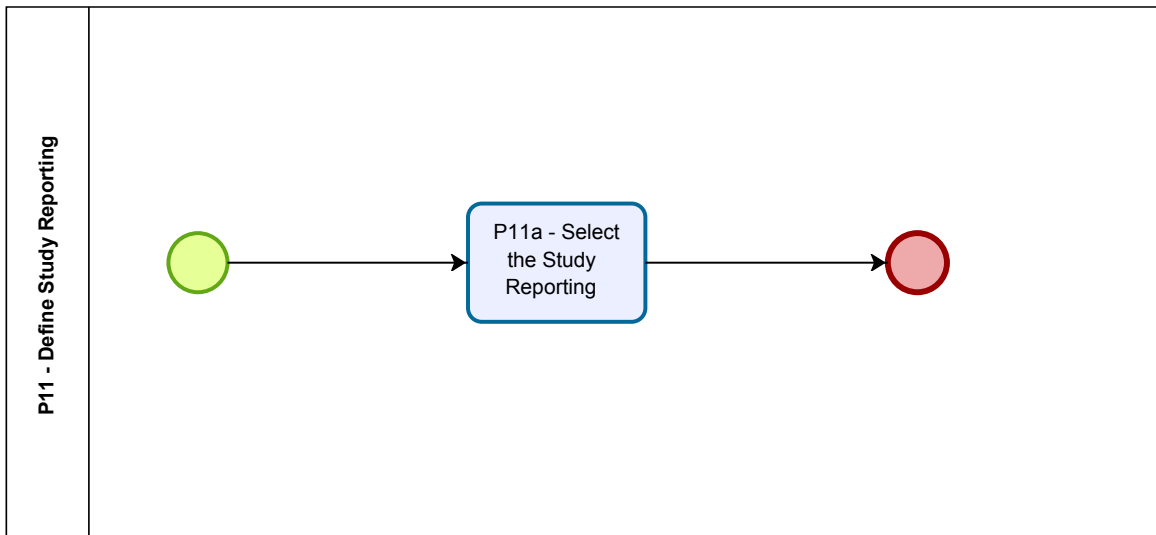


Figure C.30: Tasks P11a - Select the Study Reporting

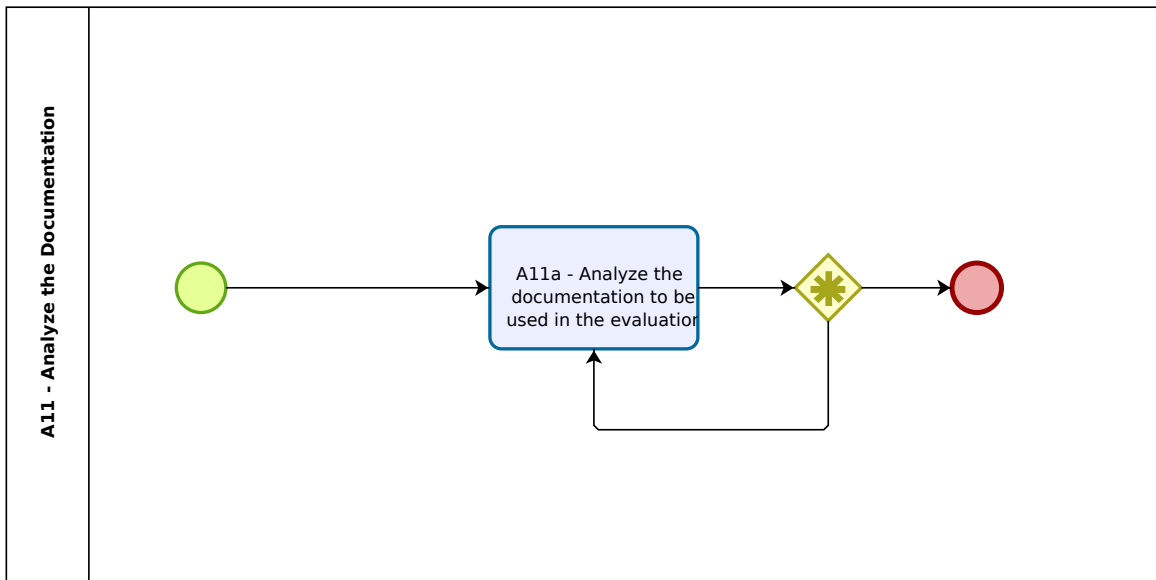


Figure C.31: Task A11a - Analyze the Documentation to be Used in the Evaluation

R11a - Present the Evaluation According to the Report Template

R11b - Present the Evaluation According to the Scientific Paper Template

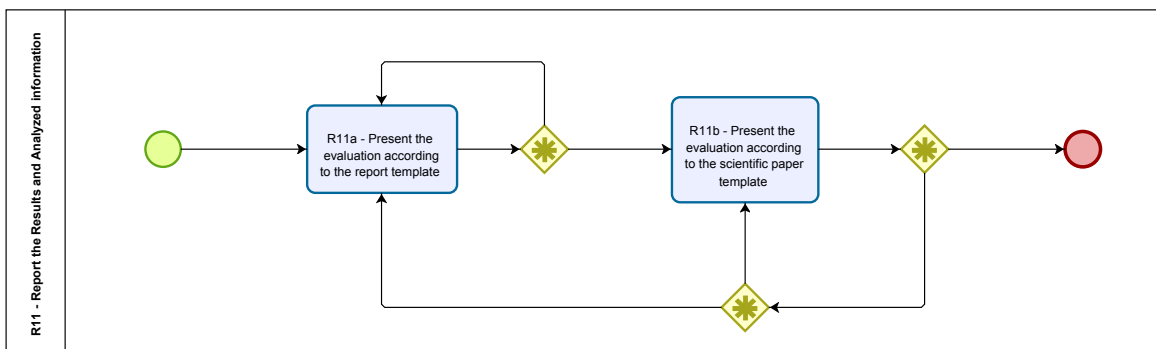


Figure C.32: Tasks of R11 - Report the Results and Analyzed Information



Heuristic Evaluation CheckList for Graphical and Textual DSL

This checklist is designed to be used together with the Process. The Nielsen Heuristics are used in the Heuristic Evaluation Checklist.

The following 0 to 4 rating scale can be used to rate the severity of usability problems:

Severity	Type	Description
0	Not applicable	I don't agree that this is a usability problem at all
1	Cosmetic problem only	need not be fixed unless extra time is available on project
2	Minor usability problem	fixing this should be given low priority
3	Major usability problem	important to fix, so should be given high priority
4	Usability catastrophe	imperative to fix this before product can be released





Heuristic	Description	Question	Apply	Not apply	Description of each error occurrence	Severity (Check each of the problems found)				
						0	1	2	3	4
H1: Visibility of system status	The DSL should always keep users informed about what is going on, through appropriate feedback within reasonable time.	Does the Graphical DSL provide immediate and adequate feedback on its status for each user action? For example, after an include or exclude task the language displays a commit message?								
		Does the Textual DSL provide immediate and adequate feedback on its status for each user action? For example, after an include or exclude task the language displays a commit message?								
		Do the elements available for the user specifically execute only one command? For example, the "undo" button only performs undo actions.								





		Do all alert and error messages that DSL (Textual or Grafica) provides stay on the screen long enough to be read more than once or (preferably) until the user chooses to close them?								
		Are the Textual and Graphical DSLs structured in a coupled way? For example, if the user changes something on the Textual DSL, this change must be observed on the Graphical DSL and if the user changes something on the Visual DSL, this change must be observed on the Textual DSL.								
H2: Match between language and the real world domain	The DSL must represent the user's language with words, phrases, and concepts familiar to the user instead of system-oriented. It should follow the conventions of the real world, making the information natural and logical.	Are the elements that compose the Graphical DSL used on the user's real world domain? For example, a DSL developed for a domain related to libraries needs to have digital representations of elements on this domain, such as books and notebooks.								
		Are the reserved keywords that compose the Textual DSL used on the user's real world domain? For example, a DSL developed for a domain related to libraries needs to have keywords related to the library								





		domain.																	
		Are the elements that compose the Graphical DSL a good representation of the DSL's domain?																	
		Are the elements that compose the Textual DSL a good representation of the DSL's domain?																	
		Do the elements that compose the Textual DSL meet the domain goal? The user understands the DSL goal by interacting with its elements.																	
		Do the elements that compose the Graphical DSL meet the domain goal? The user understands the DSL goal by interacting with its elements.																	
H3: User control and freedom	Users often choose DSL functions by mistake and will need	Does Textual DSL have redundancy in the undo / redo commands (e.g. key combination)?																	





	a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.	Does Graphical DSL have redundancy in the undo / redo commands (e.g. icon, menu)?								
H4: Consistency and standards	Users should not have to wonder whether different words, situations, or actions mean the same thing.	Are all the Graphical elements connected to each other? For example, are all the visual representations shown in the same contrast level and size?.								
		Can all the elements on the Graphical DSL be unambiguously recognized by the end users?								
		Are all the Textual elements apparently connected to each other? For example, are all the reserved keywords shown in the same color?								
		Are all the Textual elements connected to each other according to their names? For example, are all the reserved keywords written in the same style, only using camelCase or only using underscores (_) to separate								





		words?								
		In case of using abbreviations, are all the abbreviations consistent throughout the DSL usage?								
H5: Error prevention	Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.(confirmation box)	Does the Graphical DSL show confirmation boxes when the user performs actions considered dangerous (deleting things, etc)?								
		Does the Textual DSL show confirmation boxes when the user performs actions considered dangerous (deleting things, etc)?								
		Does the Textual DSL provide a mechanism in order to inform the restrictions of usage in case of, for example, mistyping or committing any kind of mistake?								
H6: Recognition rather than recall	Minimize the user's memory load by making objects, actions, and options visible. The user	Do reserved keywords have a meaningful and easy name so that the user can remember them in a fast way? In this case, keywords are words related to the domain of the textual								





	should not have to remember information from one part of the system to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.	DSL.								
		Is the Graphical DSL composed of elements easy to be seen on the screen (with good-contrast color, easily identified by the end user)? Is the Textual DSL composed of words easy to be seen on the screen (with good-contrast color, easily identified by the end user)? For example, commands like “if/else” are shown in a different color from the rest of the code.								
H7: Flexibility and efficiency of use	Accelerators — unseen by the novice user — may often speed up the interaction for the expert user such that the language can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.	Does the Graphical DSL have shortcuts for expert users to make the usage of the DSL in a fast way? For example, providing short-cuts or functions to write basic code structures.								
		Does the Textual DSL have shortcuts for expert users to make the usage of the DSL in a fast way? For example, providing short-cuts or functions to write basic code structures.								
		Does the Graphical DSL provide all the elements in order to complete the desired task?								





		Does the Textual DSL provide all the elements in order to complete the desired task?								
		Are common tasks easy and fast to be executed? For example, opening a file or deleting an element.								
H8: Aesthetic and minimalist design	Dialogues should not contain information that is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.	The colors of the DSL help to understand the current state of the system, but aren't they the only sources of information for understanding a state?								
		In case of showing a dialog, does this dialog have the information shown in a short-text? The quantity of lines of a short text may vary according to the dialog and the information being exposed, but it is advised that a dialog does not have more than 3 lines of warning.								
H9: Help users recognize, diagnose, and recover from errors	Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.	In case of error, does the DSL show an appropriate error message where the user, after reading it, knows how to recover from the error or where to look for a solution?								
		In case of showing a dialog (when the								





		user commits a mistake, for example), does this dialog show the possible reasons why the mistake was committed in a clear way?								
		Does the Textual DSL have a mechanism that highlights syntax errors in the text or reserved words?								
		Does Textual DSL have mechanisms to highlight errors of nesting or organization of reserved words that make the model inconsistent?								
		Does Graphical DSL have mechanisms that mark connection errors between elements that can make the model inconsistent?								
		Does Textual DSL have mechanisms that mark connection errors between elements that can make the model inconsistent?								
H10: Help and documentation	Even though it is better if the system can be used without documentation, it may be necessary to	Does the Graphical DSL have a tutorial in order to provide what is the DSL's goal, the representations used and its usage (providing examples to show the DSL in practice)?								





	provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.	Does the Textual DSL have a tutorial in order to provide the DSL's goal, the syntax usage and its meaning, as well as provide examples of the DSL's usage?								
		Are all the elements included on the Graphical DSL described in a clear way on the documentation support?								
		Are all the commands included on the Textual DSL described in a clear way on the documentation support?								
		Does the DSL (Textual or Graphical) have contextual help for each element?								



APPENDIX E – USABILITY QUESTIONNAIRE FOR GRAPHICAL AND TEXTUAL DSL - USA-DSL (EXTRATED AND ADAPTED THE COGNITIVE DIMENSIONS QUESTIONNAIRE)



Usa-DSL Process: Process for Usability Evaluation of Domain-Specific Languages

Usability Questionnaire for DSL

This questionnaire was developed based on the Cognitive Dimensions Questionnaire as a tool for assessing the usability of information devices by means of the Cognitive Dimensions of Notations framework (Alan F. Blackwell and Thomas R.G. Green, 2007).

For further reading on the framework, see:

<http://www.cl.cam.ac.uk/~afb21/CognitiveDimensions/>

The original version the Cognitive Dimensions Questionnaire, see:

<http://www.cl.cam.ac.uk/~afb21/CognitiveDimensions/CDquestionnaire.pdf>

This questionnaire collects your views about how easy it is to use a Domain-Specific Language (DSL). The questionnaire includes a series of questions that encourage you to think about the ways you need to use one particular notational system, and whether it helps you to do the things you need.

Part I: Definitions

You might need to think carefully to answer the questions in the next sections, so we have provided some definitions and an example to get you started:

Product: The product is the ultimate reason why you are using the notational system – what things happen as a result, or what things will be produced as a result of using the notational system. This event or object is called the product. Any product that needs a notation to describe it usually has some complex structure (Blackwell and Green, 2007).

DSL: In this context, the DSL is a language or a notation, and is how you communicate with the system – you provide information in some special format to describe the result that you want, and the notation provides information that you can read. Notations have a structure that corresponds in some way to the structure of the product they describe. They also have parts (components, aspects etc.) that correspond in some way to parts of the product. Notations can include text, pictures, diagrams, tables, special symbols or various combinations of these. Some systems include multiple notations. These might be quite similar to each other – for example when using a typewriter, the text that it produces is just letters and characters, while the notation on the keys that you press tells you exactly how to get the result you want. In other cases, a system might include some notations that are hard for humans to produce or to read. For example, when you use a telephone the notation on the buttons is a simple arrangement of digits, but the noises you hear are not so easy to interpret (different dialing tones for each number, clicks, and ringing tones). A telephone with a display therefore provides a further notation that is easier for the human user to understand (Blackwell and Green, 2007).



Sub-devices: Complex systems can include several specialized notations to help with a specific part of the job. Some of these might not normally be considered part of the system, for example, when you stick a Post-It note on your computer screen to remind you what to write in a word processor document. There are two kinds of these sub-devices.

- The Post-It note is an example of a helper device. Another example is when you make notes of telephone numbers on the back of an envelope: the complete system is the telephone plus the paper notes – if you did not have some kind of helper device like the envelope, the telephone would be much less useful.
- A redefinition device changes the main notation in some way – such as defining a keyboard shortcut, a quick-dial code on a telephone, or a macro function. The redefinition device allows you to define these shortcuts, redefine them, and delete them and so on.

Note that both helper devices and redefinition devices need their own notations that are separate from the main notation of the system. We therefore ask you to consider them separately in the rest of this questionnaire (Blackwell and Green, 2007).

Example of Use: To review how we intend to use these terms, consider the example of writing acceptance test scripts. The product of using the tester is the Cucumber. Cucumber is a software tool that supports Behavior-Driven Development (BDD). The notation in this case is the DSL that Cucumber implements. The Cucumber BDD approach fulfils a textual DSL called Gherkin. It allows expected software behaviors to be specified in a logical language that stakeholders may understand. A "Feature" is a Use Case that describes a specific function of the software being tested. Each "Feature" is made of a collection of "Scenarios". A single "Scenario" is a flow of events (Steps) through the "Feature" being described and maps 1:1 with an executable test case for the system. In some cases, one might want to test multiple "Scenarios" at once. In this case, a "Scenario Outline" provides a technique to specify multiple examples to test against a template scenario by using placeholders to be replaced by test data. The crux of a "Scenario" is defined by a sequence of "Steps" outlining the preconditions and flow of events that will take place. The first word of a step is a keyword, typically one of: "Given" - Describes the preconditions and initial state before the start of a test and allows for any pre-test setup that may occur; "When" - Describes actions taken by a user during a test; "Then" - Describes the outcome resulting from actions taken in the When clause; Occasionally, the combination of Given-When-Then uses other keywords to define conjunctions: "And" - Logical and; "But" - Logically the same as And, but used in the negative form.




▼ Feature: To search cucumber in google 1

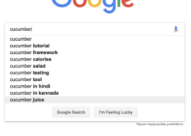
@CucumberScenario

▼ Scenario: Cucumber Google Search 4

✓ Given I am on "google" search page [Screenshot -](#) 533ms



✓ When I type "cucumber" [Screenshot -](#) 586ms



▼ Feature: To search protractor in google 1

@ProtractorScenario

▼ Scenario: Protractor Google Search 4

✓ Given I am on "cucumber" search page [Screenshot +](#) 725ms

✓ When I type "protractor" [Screenshot +](#) 846ms

✓ When I click on search button [Screenshot +](#) 1s 575ms

✓ Then I clear the search text [Screenshot +](#) 684ms

Part II: Questions about System

1) What task or activity do you use the system for?

2) What is the product of using the system?

3) What is the main notation of the system?

4) When using the system, what proportion of your time do you spend:

- a. Searching for information within the DSL: _____
- b. Translating substantial amounts of information from some other source into the system: _____
- c. Reorganizing and restructuring descriptions that you have previously created: _____



5) Are there any helper devices? Please list them here, and fill out a separate copy of section 5 for each one.

Part III: Questions about Notation/DSL

Dimensions	Questions	Answers
Visibility	How easy is it to see or find the various parts of the notation while it is being created or changed? Why?	
	What kind of things are more difficult to see or find?	
	If you need to compare or combine different parts, can you see them at the same time? If not, why not?	
Viscosity	When you need to make changes to previous work, how easy is it to make the change? Why?	
	Are there particular changes that are more difficult or especially difficult to make? Which ones?	
Diffuseness	Does the notation a) let you say what you want reasonably briefly, or b) is it long-winded? Why?	
	What sorts of things take more space to describe?	
Hard Mental Operation	What kind of things require the most mental effort with this notation?	
	Do some things seem especially complex or difficult to work out in your head (e.g. when combining several things)? What are they?	
Error-proneness	Do some kinds of mistakes seem particularly common or easy to make? Which ones?	
	Do you often find yourself making small slips that irritate you or make you feel stupid? What are some examples?	
Closeness of mapping	How closely related is the notation to the result that you are describing? Why? (Note that in a sub-device, the result may be part of another notation, rather than the end product).	
	Which parts seem to be a particularly strange way of doing or describing something?	
Role-expressiveness	When reading the notation, is it easy to tell what each part is for in the overall scheme? Why?	
	Are there some parts that are particularly difficult to interpret? Which ones?	



	Are there parts that you really do not know what they mean, but you put them in just because it has always been that way? What are they?	
Hidden dependencies	If the structure of the product means some parts are closely related to other parts, and changes to one may affect the other, are those dependencies visible? What kind of dependencies are hidden?	
	In what ways can it get worse when you are creating a particularly large description?	
	Do these dependencies stay the same, or are there some actions that cause them to get frozen? If so, what are they?	
Progressive evaluation	How easy is it to stop in the middle of creating some notation, and check your work so far? Can you do this any time you like? If not, why not?	
	Can you find out how much progress you have made, or check what stage in your work you are up to? If not, why not?	
	Can you try out partially completed versions of the product? If not, why not?	
Provisionality	Is it possible to sketch things out when you are playing around with ideas, or when you are not sure which way to proceed? What features of the notation help you to do this?	
	What sort of things can you do when you do not want to be too precise about the exact result you are trying to get?	
Premature commitment	When you are working with the notation, can you go about the job in any order you like, or does the system force you to think ahead and make certain decisions first?	
	If so, what decisions do you need to make in advance? What sort of problems can this cause in your work?	
Consistency	Where there are different parts of the notation that mean similar things, is the similarity clear from the way they appear? Please give examples.	
	Are there places where some things ought to be similar, but the notation makes them different? What are they?	
Secondary notation	Is it possible to make notes to yourself, or express information that is not really recognized as part of the notation?	
	If it was printed on a piece of paper that you could annotate or scribble on, what would you write or draw?	



	Do you ever add extra marks (or colors or format choices) to clarify, emphasize or repeat what is there already? [If yes, does this constitute a helper device? If so, please fill in one of the section 5 sheets describing it]	
Abstraction	Does the system give you any way of defining new facilities or terms within the notation, so that you can extend it to describe new things or to express your ideas more clearly or succinctly? What are they?	
	Does the system insist that you start by defining new terms before you can do anything else? What sort of things?	
	If you wrote here, you have a redefinition device: please fill in one of the section 5 sheets describing it.	

Do you find yourself using this notation in ways that are unusual, or ways that the designer might not have intended? If so, what are some examples?

After completing this questionnaire, can you think of obvious ways that the design of the system could be improved? What are they?

Part IV: Questions about Sub-devices

Please fill out a copy of this part for each sub-device in the system.

This part is describing () a helper device, or () or a redefinition device

- 1) What is its name? _____
- 2) What kind of notation is used in this sub-device?

- 3) When using this sub-device, what proportion of the time using it do you spend:
 - a. Searching for information: _____
 - b. Translating substantial amounts of information from some other source into the system: _____
 - c. Adding small bits of information to a description that you have previously created: _____
 - d. Reorganizing and restructuring descriptions that you have previously created: _____



- 4) In what ways is the notation in this sub-device different from the main notation?
Please tick boxes where there are differences, and write a few words explaining the difference.

Dimensions	Questions	Answers
Viscosity	Is it easy to see different parts?	
Visibility	Is it easy to make changes?	
Diffuseness	Is the notation succinct or long-winded?	
Hard Metal Operation	Do some things require hard mental effort?	
Error-prevention	Is it easy to make errors or slips?	
Closeness of mapping	Is the notation closely related to the result?	
Role-expressiveness	Is it easy to tell what each part is for?	
Hidden dependencies	Are dependencies visible?	
Progressive evaluation	Is it easy to stop and check your work so far?	
Provisionality	Is it possible to sketch things out?	
Premature commitment	Can you work in any order you like?	
Consistency	Are any similarities between different parts clear?	
Secondary notation	Can you make informal notes to yourself?	
Abstraction	Can you define new terms or features?	

Do you use this notation in unusual ways?

How could the design of the system be improved?

APPENDIX F – USABILITY QUESTIONNAIRE FOR GRAPHICAL AND TEXTUAL DSL - USA-DSL VERSION ADAPTED



Usa-DSL Process: Process for Usability Evaluation of Domain-Specific Languages

Usability Questionnaire for DSL - Adapted Version

This questionnaire was developed based on the Cognitive Dimensions Questionnaire as a tool for evaluating the usability of information devices by means of the Cognitive Dimensions of Notations Framework (Alan F. Blackwell and Thomas R.G. Green, 2007).

For further reading on the framework, see:

<http://www.cl.cam.ac.uk/~afb21/CognitiveDimensions/>

The original version the Cognitive Dimensions Questionnaire, see:

<http://www.cl.cam.ac.uk/~afb21/CognitiveDimensions/CDquestionnaire.pdf>

This questionnaire collects your views about how easy it is to use a Domain-Specific Language (DSL). The questionnaire includes a series of questions that encourage you to think about the ways you need to use one particular notational system, and whether it helps you to do the things you need.

Part I: Definitions

You might need to think carefully to answer the questions in the next sections, so we have provided some definitions and an example to get you started:

Product: The product is the ultimate reason why you are using the notational system – what things happen as a result, or what things will be produced as a result of using the notational system. This event or object is called the product or framework. Any product that needs a notation to describe it usually has some complex structure (Blackwell and Green, 2007).

DSL: In this context, the DSL is a language or a notation, and is how you communicate with the system – you provide information in some special format to describe the result that you want, and the notation provides information that you can read. Notations have a structure that corresponds in some way to the structure of the product they describe. They also have parts (components, aspects etc.) that correspond in some way to parts of the product. Notations can include text, pictures, diagrams, tables, special symbols or various combinations of these. Some systems include multiple notations. These might be quite similar to each other – for example when using a typewriter, the text that it produces is just letters and characters, while the notation on the keys that you press tells you exactly how to get the result you want. In other cases, a system might include some notations that are hard for humans to produce or to read. For example, when you use a telephone the notation on the buttons is a simple arrangement of digits, but the noises you hear are not so easy to interpret (different dialing tones for each number, clicks, and ringing tones). A telephone with a display therefore provides a further notation that is easier for the human user to understand (Blackwell and Green, 2007).





Sub-devices: Complex systems can include several specialized notations to help with a specific part of the job. Some of these might not normally be considered part of the system, for example, when you stick a Post-It note on your computer screen to remind you what to write in a word processor document. There are two kinds of these sub-devices.

- The Post-It note is an example of a helper device. Another example is when you make notes of telephone numbers on the back of an envelope: the complete system is the telephone plus the paper notes – if you did not have some kind of helper device like the envelope, the telephone would be much less useful.
- A redefinition device changes the main notation in some way – such as defining a keyboard shortcut, a quick-dial code on a telephone, or a macro function. The redefinition device allows you to define these shortcuts, redefine them, and delete them and so on.

Note that both helper devices and redefinition devices need their own notations that are separate from the main notation of the system. We therefore ask you to consider them separately in the rest of this questionnaire (Blackwell and Green, 2007).

Example of Use: To review how we intend to use these terms, consider the example of writing acceptance test scripts. The product of using the tester is the Cucumber. Cucumber is a software tool that supports Behavior-Driven Development (BDD). The notation in this case is the DSL that Cucumber implements. The Cucumber BDD approach fulfils a textual DSL called Gherkin. It allows expected software behaviors to be specified in a logical language that stakeholders may understand. A "Feature" is a Use Case that describes a specific function of the software being tested. Each "Feature" is made of a collection of "Scenarios". A single "Scenario" is a flow of events (Steps) through the "Feature" being described and maps 1:1 with an executable test case for the system. In some cases, one might want to test multiple "Scenarios" at once. In this case, a "Scenario Outline" provides a technique to specify multiple examples to test against a template scenario by using placeholders to be replaced by test data. The crux of a "Scenario" is defined by a sequence of "Steps" outlining the preconditions and flow of events that will take place. The first word of a step is a keyword, typically one of: "Given" - Describes the preconditions and initial state before the start of a test and allows for any pre-test setup that may occur; "When" - Describes actions taken by a user during a test; "Then" - Describes the outcome resulting from actions taken in the When clause; Occasionally, the combination of Given-When-Then uses other keywords to define conjunctions: "And" - Logical and; "But" - Logically the same as And, but used in the negative form.




▼ Feature: To search cucumber in google 1

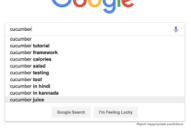
@CucumberScenario

▼ Scenario: Cucumber Google Search 4

✓ Given I am on "google" search page [Screenshot -](#) 533ms



✓ When I type "cucumber" [Screenshot -](#) 586ms



▼ Feature: To search protractor in google 1

@ProtractorScenario

▼ Scenario: Protractor Google Search 4

✓ Given I am on "cucumber" search page [Screenshot +](#) 725ms

✓ When I type "protractor" [Screenshot +](#) 846ms

✓ When I click on search button [Screenshot +](#) 1s 575ms

✓ Then I clear the search text [Screenshot +](#) 684ms

Part II: Questions about System

1) What task or activity do you use the system for?

2) What is the product of using the system?

3) What is the main notation of the system?

4) When using the system, what proportion of your time do you spend:

- a. Searching for information within the DSL: _____
- b. Translating substantial amounts of information from some other source into the system: _____
- c. Reorganizing and restructuring descriptions that you have previously created: _____



5) Are there any helper devices? Please list them here, and fill out a separate copy of section 5 for each one.

6) In your opinion, what are the positive points of the System?

7) In your opinion, what are the negative points of System?

8) Do you have any suggestions for improvement to implement in the system? (if applicable)

Part III: Questions about Notation/DSL

Questions	Strong Disagree			Strong Agree		Answers
	1	2	3	4	5	
1) How easy is DSL to use?						
2) How easy is it to see or find the various parts of the notation while it is being created or changed? Why?						
3) What kind of things are more difficult to see or find?						
4) If you need to compare or combine different parts, can you see them at the same time? If not, why not?						
When you need to make changes to previous work, how easy is it to make the change? Why?						
Are there particular changes that are more difficult or especially difficult to make? Which ones?						
Does the notation a) let you say what you want reasonably briefly, or b) is it long-winded? Why?						
What sorts of things take more space to describe?						
What kind of things require the most mental effort with this notation?						
Do some things seem especially complex or difficult to work out in your head (e.g. when combining several things)? What are they?						



Do some kinds of mistakes seem particularly common or easy to make? Which ones?						
Do you often find yourself making small slips that irritate you or make you feel stupid? What are some examples?						
How closely related is the notation to the result that you are describing? Why? (Note that in a sub-device, the result may be part of another notation, rather than the end product).						
Which parts seem to be a particularly strange way of doing or describing something?						
When reading the notation, is it easy to tell what each part is for in the overall scheme? Why?						
Are there some parts that are particularly difficult to interpret? Which ones?						
Are there parts that you really do not know what they mean, but you put them in just because it has always been that way? What are they?						
If the structure of the product means some parts are closely related to other parts, and changes to one may affect the other, are those dependencies visible? What kind of dependencies are hidden?						
In what ways can it get worse when you are creating a particularly large description?						
Do these dependencies stay the same, or are there some actions that cause them to get frozen? If so, what are they?						
How easy is it to stop in the middle of creating some notation, and check your work so far? Can you do this any time you like? If not, why not?						
Can you find out how much progress you have made, or check what stage in your work you are up to? If not, why not?						
Can you try out partially completed versions of the product? If not, why not?						
Is it possible to sketch things out when you are playing around with ideas, or when you are not sure which way to proceed? What features of the notation help you to do this?						
What sort of things can you do when you do not want to be too precise about the exact result you are trying to get?						
When you are working with the notation, can you go about the job in any order you like, or does the system force you to think ahead and make certain decisions first?						



If so, what decisions do you need to make in advance? What sort of problems can this cause in your work?						
Where there are different parts of the notation that mean similar things, is the similarity clear from the way they appear? Please give examples.						
Are there places where some things ought to be similar, but the notation makes them different? What are they?						
Is it possible to make notes to yourself, or express information that is not really recognized as part of the notation?						
If it was printed on a piece of paper that you could annotate or scribble on, what would you write or draw?						
Do you ever add extra marks (or colors or format choices) to clarify, emphasize or repeat what is there already? [If yes, does this constitute a helper device? If so, please fill in one of the section 5 sheets describing it]						
Does the system give you any way of defining new facilities or terms within the notation, so that you can extend it to describe new things or to express your ideas more clearly or succinctly? What are they?						
Does the system insist that you start by defining new terms before you can do anything else? What sort of things?						
If you wrote here, you have a redefinition device: please fill in one of the section 5 sheets describing it.						

Do you find yourself using this notation in ways that are unusual, or ways that the designer might not have intended? If so, what are some examples?

After completing this questionnaire, can you think of obvious ways that the design of the system could be improved? What are they?



Part IV: Questions about Sub-devices

Please fill out a copy of this part for each sub-device in the system.

This part is describing () a helper device, or () or a redefinition device

- 1) What is its name? _____
- 2) What kind of notation is used in this sub-device? _____
- 3) When using this sub-device, what proportion of the time using it do you spend:
 - a. Searching for information: _____
 - b. Translating substantial amounts of information from some other source into the system: _____
 - c. Adding small bits of information to a description that you have previously created: _____
 - d. Reorganizing and restructuring descriptions that you have previously created: _____

- 4) In what ways is the notation in this sub-device different from the main notation?
Please tick boxes where there are differences, and write a few words explaining the difference.

Dimensions	Questions	Answers
Viscosity	Is it easy to see different parts?	
Visibility	Is it easy to make changes?	
Diffuseness	Is the notation succinct or long-winded?	
Hard Metal Operation	Do some things require hard mental effort?	
Error-prevention	Is it easy to make errors or slips?	
Closeness of mapping	Is the notation closely related to the result?	
Role-expressiveness	Is it easy to tell what each part is for?	
Hidden dependencies	Are dependencies visible?	
Progressive evaluation	Is it easy to stop and check your work so far?	
Provisionality	Is it possible to sketch things out?	
Premature commitment	Can you work in any order you like?	
Consistency	Are any similarities between different parts clear?	



Secondary notation	Can you make informal notes to yourself?	
Abstraction	Can you define new terms or features?	

Do you use this notation in unusual ways?

How could the design of the system be improved?

APPENDIX G – INFORMED CONSENT TERM



Informed Consent Term

Subject: _____ Date: _____

You are being invited to participate in the research _____, under the responsibility of the master student/research _____, under the guidance of Professor Dr(a)_____.

You will participate in a (Empirical Study Method (SE))_____ that will review and discuss documents related to the (DSL) _____. There will be a questionnaire application, which will record your profile and collect information about your appreciation of the _____. The information obtained through this research will be confidential and we will ensure the confidentiality of your participation. Thus, the disclosed data will not allow any identification.

Your participation is voluntary and if you decide not to participate or you want to cancel your participation at any time, you have the absolute freedom to do so.

Even without having direct benefits in participating, indirectly you will be contributing to the understanding of the phenomenon studied and to the production of scientific knowledge.

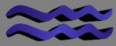
Any questions regarding the research can be made through the researchers' emails: _____.

STATEMENT OF CONSENT OF THE SUBJECT OF THE RESEARCH

I agree to participate in this study and I declare that I have read the details described in this document. I understand that I am free to accept or refuse, and that I can interrupt my participation at any time without giving a reason why. I agree that the data collected will be used for the purpose described above. I understand the information presented in this CONSENT TERMS. I had the opportunity to ask questions and all my questions were answered. I will receive a signed and dated copy of this FREE AND CLARIFIED CONSENT Document.

<p style="text-align: center;">[to be filled by the researchers]</p> <p style="text-align: center;">Special conditions (if there are no special conditions, write "none"):</p> <p>_____</p> <p>_____</p> <p>_____</p> <p>_____</p>	<p style="text-align: center;">With the knowledge of the exposed information, I express my agrément of spontaneous willingness to participate in the study.</p> <p style="text-align: center;">_____</p> <p style="text-align: center;">Signature of the participant</p> <p style="text-align: center;">_____</p> <p style="text-align: center;">Signature of the Researcher</p> <p style="text-align: center;">_____</p> <p style="text-align: center;">Signature of the Researcher</p>
---	--

APPENDIX H – PROFILE QUESTIONNAIRE



Usa-DSL Process: Process for Usability Evaluation of Domain-Specific Languages

Profile Questionnaire

Please fill all the questions in respective fields. The questions which have options with the fields "○" are exclusive selection, and questions which have options with the fields "□" is multiple choice.

1. Have you ever used a Domain Specific Language (DSL)?*
 - Yes
 - No

2. Have you ever created a Domain Specific Language (DSL)?*
 - Yes
 - No

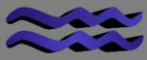
3. What empirical studies have you participated in? *
 - Case Study
 - Controlled Experiment
 - Quase-Experiment
 - Survey(Questionnaires)
 - Proof of Concept
 - I never take part in an empirical study

4. Which category (s) do you currently belong to? *
 - Academy
 - Industry

5. What is your education level? *
 - Undergraduate
 - Graduate
 - Uncompleted Master
 - Completed Master
 - Uncompleted PhD
 - Completed PhD
 - Other: _____



Usa-DSL Process ©



6. How long do you have experience in the industry? *

- Don't have
- Up to 1 year
- Between 1 year and 3 years
- Between 3 years and 6 years
- Between 6 yeras and 9 years
- Over 9 years

7. How long do you have experience in the industry? *

- Don't have
- Up to 1 year
- Between 1 year and 3 years
- Between 3 years and 6 years
- Between 6 yeras and 9 years
- Over 9 years





Pontifícia Universidade Católica do Rio Grande do Sul
Pró-Reitoria de Graduação
Av. Ipiranga, 6681 - Prédio 1 - 3º. andar
Porto Alegre - RS - Brasil
Fone: (51) 3320-3500 - Fax: (51) 3339-1564
E-mail: prograd@pucrs.br
Site: www.pucrs.br